CONTENTS

CHAPTER 1

Overview 1-1

Prime Access Registrar Hierarchy 1-2

UserLists and Groups 1-3

Profiles 1-3

Scripts 1-3

Services 1-3

Session Management Using Resource Managers 1-4

Prime Access Registrar Directory Structure 1-5

Program Flow 1-6

Scripting Points 1-6

Client Scripting 1-7

Client or NAS Scripting Points 1-7

Authentication and/or Authorization Scripting Points 1-8

Session Management 1-8

Failover by the NAS and Session Management 1-9

Cross Server Session and Resource Management 1-9

Script Processing Hierarchy 1-11

RADIUS Protocol 1-12

Steps to Connection 1-13

Types of RADIUS Messages 1-14

Packet Contents 1-14

The Attribute Dictionary 1-15

Proxy Servers 1-15

Service and Ports Used in Prime Access Registrar 1-16

Secure Shell Service 1-16

Ports 1-16

Related Documentation 21

Obtaining Documentation and Submitting a Service Request 1-21

1-21

CHAPTER 2

Using the aregcmd Commands 2-1

General Command Syntax 2-1

View-Only Administrator Mode 2-2

ViewOnly Property 2-3
CHAPTER 3

Using the Graphical User Interface 3-1

Launching the GUI 3-1
  Disabling HTTP 3-2
  Disabling HTTPS 3-2
Login Page 3-3
  Logging In 3-3
  Logging Out 3-4

Common Methodologies 3-4
  Filtering Records 3-4
  Editing Records 3-5
  Deleting Records 3-5
  Setting Record Limits per Page 3-6
  Performing Common Navigations 3-6
  Relocating Records 3-7

Dashboard 3-8
  Sessions 3-8

Configuring Cisco Prime Access Registrar 3-9
  RADIUS 3-10
    Setting Up or Changing the Radius Properties 3-11
  Profiles 3-11
    Adding Profile Details 3-12
  UserGroups 3-12
    Adding UserGroup Details 3-14
  UserList 3-14
    Adding UserList Details 3-15
  Users 3-16
    Adding User Details 3-17
  Scripts 3-18
    Adding Script Details 3-21
  Policies 3-22
    Adding Policy Details 3-22
  Services 3-22
    Simple Services 3-23
    ServiceWithRS 3-30
Contents

PEAP Service  3-34
EAP Service  3-38
Diameter Service  3-47
CommandSets  3-52
  Adding a Command Set  3-52
DeviceAccessRules  3-53
  Adding a Device Access Rule  3-53
FastRules  3-54
  Adding a Fast Rule  3-54
Replication  3-55
  Adding Replication Details  3-56
  Adding the Replication Member Details  3-57
RADIUSDictionary  3-57
  Adding RADIUS Dictionary Details  3-58
VendorDictionary  3-58
  Adding Vendor Dictionary Details  3-59
Vendor Attributes  3-60
  Adding Vendor Attributes  3-60
Vendors  3-61
  Adding Vendor Details  3-61
Translations  3-62
  Adding Translation Details  3-63
TranslationGroups  3-64
  Adding Translation Group Details  3-64
Diameter  3-65
  General  3-65
  Session Management  3-67
  Applications  3-69
  Commands  3-70
  DiameterAttributes  3-71
Advanced  3-72
  Default  3-73
  BackingStore/ServerParam  3-78
  RemoteSessionServer  3-83
  SNMP  3-85
  DDNS  3-88
  ODBC DataSources  3-89
  Log  3-90
  Ports  3-92
  Interfaces  3-93
Attribute Groups  3-94
Rules  3-95
  Setting Rules  3-96
SessionManagers  3-96
  Adding Session Manager Details  3-99
ResourceManager  3-99
  Adding Resource Manager Details  3-107
Network Resources  3-108
Clients  3-108
  Adding Client Details  3-113
Remote Servers  3-113
  LDAP  3-114
  LDAP Accounting  3-118
  ODBC/OCI  3-121
  ODBC/OCI-Accounting  3-123
  Diameter  3-125
  Others  3-128
Administration  3-134
  Administrators  3-134
    Adding Administrator Details  3-135
Statistics  3-135
  Resetting Server Statistics  3-138
  DiameterStatistics  3-138
  TACACSStatistics  3-148
Back Up and Restore  3-149
  LicenseUpload  3-150
Read-Only GUI  3-150

CHAPTER 4
Cisco Prime Access Registrar Server Objects  4-1
  Radius  4-2
  UserLists  4-3
    Users  4-4
      HiddenAttributes Property  4-5
  UserGroups  4-5
  Policies  4-6
  Clients  4-6
  Vendors  4-10
  Scripts  4-11
Contents

Services 4-12
  Types of Services 4-13
    EAP Services 4-13
    File 4-14
    Group 4-15
    Java 4-17
    LDAP 4-17
    Local 4-18
    ODBC 4-19
    ODBC-Accounting 4-19
  Prepaid Services 4-19
  RADIUS 4-20
    Radius Query 4-20
    RADIUS-Session 4-24
    Rex 4-24
  WiMAX 4-25
    Diameter 4-25
    M3UA 4-32
Session Managers 4-33
  Session Creation 4-36
  Session Notes 4-36
  Soft Group Session Limit 4-37
  Session Correlation Based on User-Defined Attributes 4-38
Resource Managers 4-38
  Types of Resource Managers 4-39
    Group-Session-Limit 4-40
    Home-Agent 4-40
    Home-Agent-IPv6 4-40
    IP-Dynamic 4-40
    IP-Per-NAS-Port 4-41
    IPX-Dynamic 4-41
    Session-Cache 4-41
    Subnet-Dynamic 4-42
    User-Session-Limit 4-43
    USR-VPN 4-43
    Dynamic-DNS 4-44
    Remote-IP-Dynamic 4-44
    Remote-User-Session-Limit 4-44
    Remote-Group-Session-Limit 4-44
    Remote-Session-Cache 4-44
Contents

3GPP 4-44
Profiles 4-45
Attributes 4-45
Translations 4-46
TranslationGroups 4-47
Remote Servers 4-47
Types of Protocols 4-48
  Dynamic DNS 4-49
  LDAP 4-50
  Map-Gateway 4-52
  Sigtran 4-53
  ODBC 4-54
  ODBC-Accounting 4-56
  OCI 4-56
  OCI-Accounting 4-57
  Prepaid-CRB 4-57
  Prepaid-IS835C 4-57
  RADIUS 4-57
  SIGTRAN-M3UA 4-58
Rules 4-58
Advanced 4-58
  RemoteODBCSessionServer 4-73
  Using the RequireNASsBehindProxyBelClientList Property 4-75
  Advance Duplicate Detection Feature 4-75
  Invalid EAP Packet Processing 4-75
Ports 4-76
Interfaces 4-76
Reply Messages 4-76
Attribute Dictionary 4-78
  Types 4-79
  Vendor Attributes 4-79
SNMP 4-79
Diameter 4-80
  Configuring Diameter Transport Management Properties 4-81
  Configuring Diameter Session Management 4-83
  Configuring Diameter Application 4-84
  Configuring Diameter Commands 4-85
  Configuring Diameter Dictionary 4-91
### Contents

- Configuring Profiles to Group Attributes 6-8
- Configuring Return Attributes and Check-Items Using UserGroup 6-9
- Return Attribute Precedence 6-10
- aregcmd Command Performance 6-10
- UserDefined1 Property 6-11
- Access-Request Logging 6-11

#### CHAPTER 7

**RADIUS Accounting** 7-1

- Understanding RADIUS Accounting 7-1
- Setting Up Accounting 7-2
  - Accounting Log File Rollover 7-2
    - FilenamePrefix 7-3
    - MaxFileSize 7-4
    - MaxFileAge 7-4
    - RolloverSchedule 7-5
    - UseLocalTimeZone 7-5
    - FileType 7-5
    - EnableRolloverIntelligence 7-5
    - AttributesToBeLogged 7-6
- Oracle Accounting 7-6
  - Configuring Oracle Accounting 7-7
    - ODBC-Accounting Service 7-7
    - ODBC RemoteServers 7-7
    - Configuration Examples 7-9
- Packet Buffering 7-10
  - When Using Packet Buffering 7-11
  - With Packet Buffering Disabled 7-11
- Dynamic SQL Feature 7-11
- LDAP Accounting 7-12
  - Configuring LDAP Accounting 7-12
    - LDAP-Accounting Service 7-12
    - LDAP RemoteServers 7-13
    - Configuration Examples 7-15
    - Configuring the LDAP Service for Accounting 7-16
    - Configuring an LDAP-Accounting RemoteServer 7-17
    - Setting LDAP-Accounting As Accounting Service 7-19
- MySQL Support 7-20
  - Configuring MySQL 7-20
  - Example Configuration 7-21
Contents

Proxying Accounting Records 7-21
  Configuring the Local Cisco Prime Access Registrar Server 7-22
  Configuring the Local Accounting Service 7-22
  Configuring the Remote Accounting Service 7-22
  Configuring the Group Accounting Service 7-23
  Configuring the RemoteServer Object 7-23
Accounting Log Examples 7-24
  Accounting-Start Packet 7-24
  Accounting Stop Packet 7-24
  Trace of Successful Accounting 7-24
Sample Error Messages 7-25

CHAPTER 8

Diameter 8-1
  Diameter with EAP Support 8-2
    Advertising Application Support 8-2
    Diameter EAP Conversation Flow 8-2
  Diameter Server Startup Log 8-3
  Diameter Stack Level Messages 8-4
    Capabilities Exchange Message 8-5
    Watchdog Message 8-6
    Disconnect Message 8-6
  Configuring Authentication and Authorization for Diameter 8-6
    Configuring Local Authentication and Authorization 8-6
      Configuring a Local Service and UserList 8-7
      Configuring External Authentication Service 8-8
  Configuring the Diameter Application in Prime Access Registrar 8-8
    Configuring the Transport Management Properties 8-9
    Registering Applications IDs 8-10
    Configuring the Diameter Peers 8-11
    Configure the Diameter Service 8-12
  Writing Diameter Application in Prime Access Registrar 8-15
    Configuring rex script/service for Diameter 8-16
    Scripting in Diameter 8-16
    Diameter Environment Variables 8-17
    Sample rex script/service 8-17
    Traces/Logs 8-18
  Translation Framework for Diameter 8-19
  TLS Support for Diameter 8-20
Extensible Authentication Protocols  9-1

EAP-AKA  9-2
  Configuring EAP-AKA  9-2
  Testing EAP-AKA with radclient  9-6
EAP-AKA-Prime (EAP-AKA')  9-6
  Configuring EAP-AKA'  9-7
  Testing EAP-AKA' with radclient  9-8
EAP-FAST  9-8
  Configuring EAP-FAST  9-9
  EAP-FAST Keystores  9-12
  Testing EAP-FAST with radclient  9-13
    PAC Provisioning  9-14
    Authentication  9-14
  Parameters Used for Certificate-Based Authentication  9-15
  radclient Command Reference  9-16
PAC—Credential Export Utility  9-18
  PAC Export  9-18
  PAC Display  9-19
  Syntax Summary  9-19
EAP-GTC  9-19
  Configuring EAP-GTC  9-19
  Testing EAP-GTC with radclient  9-20
EAP-LEAP  9-21
  Configuring EAP-LEAP  9-21
EAP-MD5  9-22
  Configuring EAP-MD5  9-22
EAP-Negotiate  9-23
  Configuring EAP-Negotiate  9-23
  Negotiating PEAP Tunnel Services  9-24
  Testing EAP-Negotiate with radclient  9-24
EAP-MSChapV2  9-24
  Configuring EAP-MSChapV2  9-24
  Testing EAP-MSChapV2 with radclient  9-25
EAP-SIM  9-26
  Configuring EAP-SIM  9-26
  Quintets to Triplets Conversion  9-30
## CHAPTER 10
### Using WiMAX in Cisco Prime Access Registrar

- **WiMAX - An Overview**  10-1
- **WiMAX in Cisco Prime Access Registrar**  10-2
  - Direct Interaction Between the ASN GW and Cisco Prime Access Registrar  10-3
  - Interaction Between ASN GW and Cisco Prime Access Registrar Through HA  10-6
  - Prepaid and Hot-Lining  10-7
- **Configuring WiMAX in Cisco Prime Access Registrar**  10-7
  - Configuring the Resource Manager for WiMAX  10-8
  - Configuring the Session Manager for WiMAX  10-9
  - Configuring the Query Service for WiMAX  10-9
  - Configuring WiMAX  10-10
- **WiMAX - OMA-DM Provisioning Support with BEK Key**  10-11

## CHAPTER 11
### Using Extension Points

- **Determining the Goal of the Script**  11-1
- **Writing the Script**  11-2
  - Choosing the Type of Script  11-3
    - Request Dictionary Script  11-3
    - Response Dictionary Script  11-4
    - Environment Dictionary Script  11-4
  - Adding the Script Definition  11-5
  - Adding the Example Script Definition  11-5
  - Choosing the Scripting Point  11-6
  - Testing the Script  11-6
- **About the Tcl/Tk 8.3 Engine**  11-6
- **Cisco Prime Access Registrar Scripts**  11-6
  - ACMEOutgoingScript  11-8
  - AltigaIncomingScript  11-8
  - AltigaOutgoingScript  11-8
  - ANAAAO outgoing  11-8
  - AscendIncomingScript  11-8
  - AscendOutgoingScript  11-9
  - AuthorizePPP  11-9
  - AuthorizeService  11-9
  - AuthorizeSLIP  11-9
  - AuthorizeTelnet  11-9
  - CabletronIncoming  11-9
  - CabletronOutgoing  11-9
  - CiscoIncoming  11-9
  - CiscoPremium  11-9
CiscoOutgoing  11-10
CiscoWithODAPIncomingScript  11-10
ExecCLIDRule  11-10
ExecDNISRule  11-10
ExecFilterRule  11-10
ExecNASIPRule  11-10
ExecRealmRule  11-10
ExecTimeRule  11-11
LDAPOutage  11-11
MapSourceIPAddress  11-11
ParseAAARealm  11-11
ParseAAASRealm  11-12
ParseAARealm  11-12
ParseAASRealm  11-12
ParseProxyHints  11-12
ParseServiceAndAAARealmHints  11-12
ParseServiceAndAAASRealmHints  11-12
ParseServiceAndAARealmHints  11-13
ParseServiceAndAASRealmHints  11-13
ParseServiceAndProxyHints  11-13
ParseServiceHints  11-13
ParseTranslationGroupsByCLID  11-13
ParseTranslationGroupsByDNIS  11-13
ParseTranslationGroupsByRealm  11-14
UseCLIDAsSessionKey  11-14
USRIncomingScript  11-14
USRIncomingScript-IgnoreAccountingSignature  11-14
USROutgoingScript  11-14
Internal Scripts  11-14

Extension Points in Cisco Prime Access Registrar  11-16

CHAPTER  12

Using Replication  12-1

Replication Overview  12-1
How Replication Works  12-2
Replication Data Flow  12-3
  Master Server  12-3
  Slave Server  12-3
Security  12-4
Replication Archive  12-4
Ensuring Data Integrity 12-4
  Transaction Data Verification 12-4
  Transaction Order 12-5
  Automatic Resynchronization 12-5
Full Resynchronization 12-5
Understanding Hot-Configuration 12-6
Replication’s Impact on Request Processing 12-6
Replication Configuration Settings 12-6
  RepType 12-7
  RepTransactionSyncInterval 12-7
    Master 12-7
    Slave 12-7
  RepTransactionArchiveLimit 12-8
  ReplIPAddress 12-8
  RepPort 12-8
  RepSecret 12-8
  RepIsMaster 12-9
  RepMasterIPAddress 12-9
  RepMasterPort 12-9
  Rep Members Subdirectory 12-9
  Rep Members/Slave1 12-9
  Name 12-9
  IPAddress 12-9
  Port 12-10
Setting Up Replication 12-10
  Configuring The Master 12-10
  Configuring The Member 12-11
  Verifying the Configuration 12-12
Replication Example 12-13
  Adding a User 12-13
    Master Server’s Log 12-13
    Member Server’s Log 12-13
  Verifying Replication 12-14
    Master Server’s Log 12-14
    Member Server’s Log 12-14
  Using aregcmd -pf Option 12-14
    Master Server’s Log 12-15
    Member Server’s Log 12-15
  An Automatic Resynchronization Example 12-16
CHAPTER 13
Using On-Demand Address Pools 13-1
Cisco-Incoming Script 13-3
CiscoWithODAPIncomingScript 13-3
Vendor Type CiscoWithODAP 13-4
Configuring Cisco Prime Access Registrar to Work with ODAP 13-5
Configuring Prime Access Registrar to work with ODAP 13-5
Configuring the ODAP Detailed Instructions 13-5
Setting Up an ODAP UserList 13-5
Adding ODAP Users 13-6
Setting Up an ODAP-Users Service 13-7
Setting Up an ODAP Accounting Service 13-8
Adding Session Managers 13-8
Setting Up Resource Managers 13-9
Configuring Session Managers 13-14
Configure Clients 13-15
Save Your Configuration 13-16

CHAPTER 14
Using Identity Caching 14-1
Overview 14-1
Identity Caching Features 14-2
Configuring Cisco Prime Access Registrar for Identity Caching 14-3
Starting Identity Caching 14-6
XML Interface 14-8

CHAPTER 15
Using Trusted ID Authorization with SESM 15-1
Trusted ID Operational Overview 15-1
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>15-2</td>
</tr>
<tr>
<td>Request Processing</td>
<td>15-2</td>
</tr>
<tr>
<td>Session Cache Life Cycle</td>
<td>15-3</td>
</tr>
<tr>
<td>Configuration Restrictions</td>
<td>15-3</td>
</tr>
<tr>
<td>Software Requirements</td>
<td>15-4</td>
</tr>
<tr>
<td>Installing Cisco Prime Access Registrar</td>
<td>15-4</td>
</tr>
<tr>
<td>Running the TrustedIdInstall Program</td>
<td>15-4</td>
</tr>
<tr>
<td>Using the TrustedIdInstall.bin GUI</td>
<td>15-4</td>
</tr>
<tr>
<td>Using the TrustedIdInstall Command Line</td>
<td>15-8</td>
</tr>
<tr>
<td>Configuring Cisco Prime Access Registrar for Trusted Identity with SESM</td>
<td>15-12</td>
</tr>
<tr>
<td>Configuring the RADIUS Ports</td>
<td>15-12</td>
</tr>
<tr>
<td>Configuring NAS Clients</td>
<td>15-13</td>
</tr>
<tr>
<td>Configuring AAA and SPE Services</td>
<td>15-13</td>
</tr>
<tr>
<td>Configuration Imported by TrustedIdInstall Program</td>
<td>15-13</td>
</tr>
<tr>
<td>/Radius</td>
<td>15-14</td>
</tr>
<tr>
<td>/radius/services/spe</td>
<td>15-14</td>
</tr>
<tr>
<td>/radius/services/trusted-id</td>
<td>15-14</td>
</tr>
<tr>
<td>/Radius/SessionManagers/session-cache/</td>
<td>15-14</td>
</tr>
<tr>
<td>/radius/ResourceManagers/session-cache</td>
<td>15-14</td>
</tr>
<tr>
<td>/radius/advanced/</td>
<td>15-14</td>
</tr>
<tr>
<td>/Radius/Scripts/ChangeServiceType</td>
<td>15-15</td>
</tr>
<tr>
<td>Configuring EAP-MD5 Authentication</td>
<td>15-15</td>
</tr>
<tr>
<td>Creating the CheckEap.tcl Script</td>
<td>15-15</td>
</tr>
<tr>
<td>Adding the CheckEap.tcl Script</td>
<td>15-16</td>
</tr>
<tr>
<td>Using the CheckEap.tcl Script</td>
<td>15-16</td>
</tr>
<tr>
<td>Adding the EAP-MD5 Authentication Service</td>
<td>15-17</td>
</tr>
<tr>
<td>Adding an LDAP Remote Server</td>
<td>15-17</td>
</tr>
<tr>
<td>Adding an LDAP Service</td>
<td>15-18</td>
</tr>
<tr>
<td>Saving the Configuration and Reloading the Server</td>
<td>15-19</td>
</tr>
<tr>
<td>Cisco SSG VSAs in Cisco Prime Access Registrar Dictionary</td>
<td>15-20</td>
</tr>
<tr>
<td><strong>CHAPTER 16</strong></td>
<td><strong>16-1</strong></td>
</tr>
<tr>
<td><strong>Using Prepaid Billing</strong></td>
<td><strong>16-2</strong></td>
</tr>
<tr>
<td>Overview</td>
<td>16-2</td>
</tr>
<tr>
<td>IS835C Prepaid Billing</td>
<td>16-2</td>
</tr>
<tr>
<td>Configuring IS835C Prepaid Billing</td>
<td>16-3</td>
</tr>
<tr>
<td>Setting Up a Prepaid Billing RemoteServer</td>
<td>16-3</td>
</tr>
<tr>
<td>Setting Up an IS835C Prepaid Service</td>
<td>16-4</td>
</tr>
<tr>
<td>Setting Up Local Authentication</td>
<td>16-5</td>
</tr>
<tr>
<td>Setting Up an Authentication Group Service</td>
<td>16-5</td>
</tr>
</tbody>
</table>
CRB Prepaid Billing  16-7
   Configuring CRB Prepaid Billing  16-8
      Setting Up a Prepaid Billing RemoteServer  16-8
      Setting Up a CRB Prepaid Service  16-9
      Setting Up a Local Accounting Service  16-11
      Setting Up a Local Authentication Service  16-12
      Setting Up a Prepaid Accounting Group Service  16-13
      Setting Up an Authentication Group Service  16-14
   Configuring CRB Prepaid Billing for SSG  16-15
   Generic Call Flow  16-18
      Access-Request (Authentication)  16-19
      Access-Accept (Authentication)  16-20
      Access-Request (Authorization)  16-20
      Access-Accept (Authorization)  16-21
      Accounting-Start  16-22
   Data Flow  16-22
      Access-Request (Quota Depleted)  16-22
      Accept-Accept (Quota Depleted)  16-23
      Accounting Stop (Session End)  16-23
      Accounting Response (Final Status)  16-23
   Vendor-Specific Attributes  16-25
   Implementing the Prepaid Billing API  16-27

CHAPTER  17  Using Cisco Prime Access Registrar Server Features  17-1
   Incoming Traffic Throttling  17-2
      MaximumIncomingRequestRate  17-2
      MaximumOutstandingRequests  17-2
   Backing Store Parsing Tool  17-3
   Configurable Worker Threads Enhancement  17-4
   Session-Key Lookup  17-5
   Query-Notify  17-6
      Call Flow  17-7
         Configuration Examples  17-8
         Memory and Performance Impact  17-9
   Support for Windows Provisioning Service  17-9
      Call Flow  17-10
         Example Configuration  17-10
            Environment Variables  17-11
            Master URL Fragments  17-11
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unsupported Features</td>
<td>17-12</td>
</tr>
<tr>
<td>Account Expiration and Renewal</td>
<td>17-12</td>
</tr>
<tr>
<td>Password Changing and Force Update</td>
<td>17-13</td>
</tr>
<tr>
<td>Command Completion</td>
<td>17-13</td>
</tr>
<tr>
<td>Service Grouping Feature</td>
<td>17-14</td>
</tr>
<tr>
<td>Configuration Example - AccountingGroupService</td>
<td>17-14</td>
</tr>
<tr>
<td>Summary of Events</td>
<td>17-17</td>
</tr>
<tr>
<td>Configuration Example 2 - AuthenticationGroupService</td>
<td>17-17</td>
</tr>
<tr>
<td>Summary of Events</td>
<td>17-20</td>
</tr>
<tr>
<td>SHA-1 Support for LDAP-Based Authentication</td>
<td>17-21</td>
</tr>
<tr>
<td>Remote LDAP Server Password Encryption</td>
<td>17-21</td>
</tr>
<tr>
<td>Dynamic Password Encryption</td>
<td>17-22</td>
</tr>
<tr>
<td>Logs</td>
<td>17-23</td>
</tr>
<tr>
<td>Dynamic Attributes</td>
<td>17-23</td>
</tr>
<tr>
<td>Object Properties with Dynamic Support</td>
<td>17-23</td>
</tr>
<tr>
<td>Dynamic Attribute Format</td>
<td>17-25</td>
</tr>
<tr>
<td>Tunneling Support Feature</td>
<td>17-25</td>
</tr>
<tr>
<td>Configuration</td>
<td>17-26</td>
</tr>
<tr>
<td>Example</td>
<td>17-26</td>
</tr>
<tr>
<td>Notes</td>
<td>17-26</td>
</tr>
<tr>
<td>Validation</td>
<td>17-26</td>
</tr>
<tr>
<td>xDSL VPI/VCI Support for Cisco 6400</td>
<td>17-27</td>
</tr>
<tr>
<td>Using User-Name/User-Password for Each Cisco 6400 Device</td>
<td>17-27</td>
</tr>
<tr>
<td>Format of the New User-Name Attribute</td>
<td>17-27</td>
</tr>
<tr>
<td>Apply Profile in Cisco Prime Access Registrar Database to Directory Users</td>
<td>17-28</td>
</tr>
<tr>
<td>User-Profile</td>
<td>17-28</td>
</tr>
<tr>
<td>User-Group</td>
<td>17-29</td>
</tr>
<tr>
<td>Example User-Profile and User-Group Attributes in Directory User Record</td>
<td>17-29</td>
</tr>
<tr>
<td>Directory Multi-Value Attributes Support</td>
<td>17-29</td>
</tr>
<tr>
<td>MultiLink-PPP (ML-PPP)</td>
<td>17-30</td>
</tr>
<tr>
<td>Dynamic Updates Feature</td>
<td>17-31</td>
</tr>
<tr>
<td>NAS Monitor</td>
<td>17-32</td>
</tr>
<tr>
<td>Automatic Information Collection (arbug)</td>
<td>17-33</td>
</tr>
<tr>
<td>Running arbug</td>
<td>17-33</td>
</tr>
<tr>
<td>Files Generated</td>
<td>17-33</td>
</tr>
<tr>
<td>Simultaneous Terminals for Remote Demonstration</td>
<td>17-34</td>
</tr>
<tr>
<td>Support for RADIUS Check Item Attributes</td>
<td>17-34</td>
</tr>
<tr>
<td>Configuring Check Items</td>
<td>17-34</td>
</tr>
</tbody>
</table>
Contents

User-Specific Attributes 17-35
Packet of Disconnect 17-36
  Configuring Packet of Disconnect 17-36
    Configuring the Client Object 17-36
    Configuring a Resource Manager for POD 17-37
  Proxying POD Requests from External Servers 17-38
CLI Options for POD 17-38
  query-sessions 17-38
  release-sessions 17-39
Configuring Change of Authorization Requests 17-39
  Configuring the Client Object 17-40
Dynamic DNS 17-41
  Configuring Dynamic DNS 17-42
  Testing Dynamic DNS with radclient 17-43
Dynamic Service Authorization Feature 17-44
  Configuring Dynamic Service Authorization Feature 17-44
    Setting Up the Environment Variable 17-45
Remote Session Management 17-47
Wx Interface Support for SubscriberDB Lookup 17-48
  Configuration Examples 17-48
Smart Grid Solution Management 17-50
Lawful Interception (LI) Support in Prime Access Registrar 17-50
  Configuring Lawful Intercept 17-56
TACACS+ Support for AAA 17-57

CHAPTER 18

Directing RADIUS Requests 18-1
Configuring Policies and Rules 18-1
  Configuring Policies 18-1
  Configuring Rules 18-2
  Wildcard Support 18-2
  Script and Attribute Requirements 18-3
  Validation 18-4
  Known Anomalies 18-4
Routing Requests 18-4
  Routing Requests Based on Realm 18-4
  Routing Requests Based on DNIS 18-5
  Routing Requests Based on CLID 18-6
  Routing Requests Based on NASIP 18-7
Routing Requests Based on User-Name Prefix 18-8
Attribute Translation 18-9
Translations 18-9
TranslationGroups 18-9
Parsing Translation Groups 18-10
Time of Day Access Restrictions 18-11
Setting Time Ranges in ExecTimeRule 18-12
ExecTimeRule Example Configuration 18-12
Reducing Overhead Using Policies to Group Rules 18-13
Standard Scripts Used with Rules 18-15
ExecRealmRule 18-15
ExecDNISRule 18-16
ExecCLIDRule 18-16
ExecNASIPRule 18-17
ExecPrefixRule 18-17
ExecSuffixRule 18-18
Configuring Suffix and Prefix Policies 18-19
ExecTimeRule 18-20
ParseTranslationGroupsByRealm 18-20
ParseTranslationGroupsByDNIS 18-20
ParseTranslationGroupsByCLID 18-21
ParseTranslationGroupsByDNIS 18-21

CHAPTER 19
Using FastRules to Process Packet Flow 19-1
Configuring FastRules 19-2

CHAPTER 20
Wireless Support 20-1
Mobile Node-Home Agent Shared Key 20-1
Use Case Example 20-1
Configuring User Attributes 20-2
3GPP2 Home Agent Support 20-3
Home-Agent Resource Manager 20-3
Load Balancing 20-3
Querying and Releasing Sessions 20-4
Access Request Requirements 20-5
New 3GPP2 VSAs in the Cisco Prime Access Registrar Dictionary 20-5
Session Correlation Based on User-Defined Attributes 20-5
Managing Multiple Accounting Start/Stop Messages 20-6
NULL Password Support 20-6
CHAPTER 21 Using LDAP

Configuring LDAP
- Configuring the LDAP Service
- MultipleServersPolicy
- RemoteServers
- Configuring an LDAP RemoteServer
  - DNS Look Up and LDAP Rebind Interval
  - LDAPToRadiusMappings
  - LDAPToEnvironmentMappings
  - LDAPToCheckItemMappings
- Setting LDAP As Authentication and Authorization Service
- Saving Your Configuration
- CHAP Interoperability with LDAP
- Allowing Special Characters in LDAP Usernames
- Dynamic LDAP Search Base
- Analyzing LDAP Trace Logs
- Successful Bind Message
- Bind Failure Messages
- Login Failure Messages
- Bind-Based Authentication for LDAP

CHAPTER 22 Using Open Database Connectivity

Oracle Software Requirements

Configuring ODBC/OCI
- Configuring an ODBC/OCI Service
- Configuring an ODBC/OCI RemoteServer
- ODBC Data Source
Cisco Prime Access Registrar 7.0.1 User Guide

Contents

Diameter SNMP and Statistics Support 24-3
TACACS+ SNMP and Statistics Support 24-3

SNMP Traps 24-3
  Supported Traps 24-4
    carServerStart 24-4
    carServerStop 24-4
    carInputQueueFull 24-4
    carInputQueueNotVeryFull 24-5
    carOtherAuthServerNotResponding 24-5
    carOtherAuthServerResponding 24-5
    carOtherAccServerNotResponding 24-6
    carOtherAccServerResponding 24-6
    carAccountingLoggingFailure 24-6
    carLicenseUsage 24-7
    carSigtranLicenseUsage 24-7
    carDiameterPeerDown 24-7
    carDiameterPeerUp 24-7
  Configuring Traps 24-7
    SNMP Configuration 24-8
    Configuring Trap Recipient 24-8
    Community String 24-8

CHAPTER 25
  Enforcement of Licensing Models 25-1
    TPS Licensing Features 25-1
      Enforcement Rules 25-1
      Notification Logs 25-2
      Notification - SNMP Traps 25-2
    TPS Logging Feature 25-3
    Concurrent Session License Features 25-3
      Sessions Enforcement Rules 25-4
      Notification Logs 25-4
      Notification - SNMP Traps 25-5
      Session Logging Feature 25-5

CHAPTER 26
  Backing Up the Database 26-1
    Configuration 26-1
      Command Line Utility 26-1
    Recovery 26-2
    mcdshadow Command Files 26-2
## Appendix A

### Cisco Prime Access Registrar Tcl, REX, and Java Dictionaries

<table>
<thead>
<tr>
<th>Component</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tcl Attribute Dictionaries</td>
<td>A-1</td>
</tr>
<tr>
<td>Attribute Dictionary Methods</td>
<td>A-1</td>
</tr>
<tr>
<td>Tcl Environment Dictionary</td>
<td>A-4</td>
</tr>
<tr>
<td>REX Attribute Dictionary</td>
<td>A-5</td>
</tr>
<tr>
<td>Attribute Dictionary Methods</td>
<td>A-5</td>
</tr>
<tr>
<td>REX Environment Dictionary</td>
<td>A-11</td>
</tr>
<tr>
<td>REX Environment Dictionary Methods</td>
<td>A-11</td>
</tr>
<tr>
<td>Java Attribute Dictionary</td>
<td>A-13</td>
</tr>
<tr>
<td>Java Attribute Dictionary Methods</td>
<td>A-13</td>
</tr>
<tr>
<td>Java Environment Dictionary</td>
<td>A-16</td>
</tr>
<tr>
<td>Java Environment Dictionary Methods</td>
<td>A-16</td>
</tr>
<tr>
<td>Interface Extension</td>
<td>A-17</td>
</tr>
<tr>
<td>Interface Extension Methods</td>
<td>A-18</td>
</tr>
<tr>
<td>Interface Extension for Session</td>
<td>A-18</td>
</tr>
<tr>
<td>Interface Extension for Session Methods</td>
<td>A-19</td>
</tr>
<tr>
<td>Interface Extension with initialization</td>
<td>A-19</td>
</tr>
<tr>
<td>Interface Extension with initialization Methods</td>
<td>A-20</td>
</tr>
<tr>
<td>Interface Extension for Session with initialization</td>
<td>A-20</td>
</tr>
<tr>
<td>Interface Extension for Session with initialization Methods</td>
<td>A-20</td>
</tr>
<tr>
<td>Interface Marker Extension</td>
<td>A-20</td>
</tr>
<tr>
<td>Variables in the Marker Extension Interface</td>
<td>A-21</td>
</tr>
<tr>
<td>Class Session record</td>
<td>A-24</td>
</tr>
<tr>
<td>Session Record Methods</td>
<td>A-24</td>
</tr>
</tbody>
</table>

## Appendix B

### Environment Dictionary

<table>
<thead>
<tr>
<th>Component</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco Prime Access Registrar Environment Dictionary Variables</td>
<td>B-1</td>
</tr>
<tr>
<td>Accepted-Profiles</td>
<td>B-1</td>
</tr>
<tr>
<td>Accounting-Service</td>
<td>B-2</td>
</tr>
<tr>
<td>Acquire-Dynamic-DNS</td>
<td>B-2</td>
</tr>
<tr>
<td>Acquire-Group-Session-Limit</td>
<td>B-2</td>
</tr>
<tr>
<td>Acquire-Home-Agent</td>
<td>B-2</td>
</tr>
<tr>
<td>Acquire-IP-Dynamic</td>
<td>B-2</td>
</tr>
<tr>
<td>Acquire-IPX-Dynamic</td>
<td>B-2</td>
</tr>
<tr>
<td>Acquire-IP-Per-NAS-Port</td>
<td>B-2</td>
</tr>
<tr>
<td>Acquire-Subnet-Dynamic</td>
<td>B-3</td>
</tr>
</tbody>
</table>
Acquire-User-Session-Limit B-3
Acquire-USR-VPN B-3
Allow-Null-Password B-3
Authentication-Service B-3
Authorization-Service B-3
AuthorizationInfo B-3
BackingStore-Env-Vars B-4
Blacklisted-IMSI B-4
Broadcast-Accounting-Packet B-4
Cache-Attributes-In-Session B-4
Current-Group-Count B-4
Cache-Outer-Identity B-4
Destination-IP-Address B-4
Destination-Port B-4
Dest-Translation-Type B-5
Dest-Numbering-Plan B-5
Dest-Encoding-Scheme B-5
Dest-Nature-Of-Address B-6
Dest-GT-Format B-6
Diameter-Application-Id B-6
Diameter-Command-Code B-6
Disable-Accounting-On-Off-Broadcast B-7
DSA-Response-Cache B-7
Dynamic-DNS-HostName B-7
Dynamic-Search-Filter B-7
Dynamic-Search-Path B-7
Dynamic-Search-Scope B-7
Dynamic-Service-Loop-Limit B-7
Dynamic-User-Password-Attribute B-7
EAP-Actual-Identity B-8
EAP-Authentication-Mode B-8
Enforce-Traffic-Throttling B-8
FetchAuthorizationInfo B-8
Generate-BEK B-8
Group-Session-Limit B-8
HLR-GlobalTitle-Address B-8
HLR-GlobalTitle-Cached B-8
HLR-Translated-IMSI B-9
Ignore-Accounting-Signature B-9
IMSI B-9
<table>
<thead>
<tr>
<th>Feature</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incoming-Translation-Groups</td>
<td>B-9</td>
</tr>
<tr>
<td>Master-URL-Fragment</td>
<td>B-9</td>
</tr>
<tr>
<td>Misc-Log-Message-Info</td>
<td>B-10</td>
</tr>
<tr>
<td>MSISDN</td>
<td>B-10</td>
</tr>
<tr>
<td>Notification-Code</td>
<td>B-10</td>
</tr>
<tr>
<td>Notification-Service</td>
<td>B-10</td>
</tr>
<tr>
<td>Outgoing-Translation-Groups</td>
<td>B-10</td>
</tr>
<tr>
<td>Pager</td>
<td>B-10</td>
</tr>
<tr>
<td>PoD/CoA</td>
<td>B-11</td>
</tr>
<tr>
<td>Query-Service</td>
<td>B-11</td>
</tr>
<tr>
<td>Re-Accounting-Service</td>
<td>B-11</td>
</tr>
<tr>
<td>Re-Authentication-Service</td>
<td>B-11</td>
</tr>
<tr>
<td><strong>Re-Authorization-Service</strong></td>
<td>B-11</td>
</tr>
<tr>
<td>Reject-Reason</td>
<td>B-12</td>
</tr>
<tr>
<td>Remote-Server</td>
<td>B-12</td>
</tr>
<tr>
<td>Remove-Session-On-Acct-Stop</td>
<td>B-12</td>
</tr>
<tr>
<td>Remote-Servers-Tried</td>
<td>B-12</td>
</tr>
<tr>
<td>Request-Authenticator</td>
<td>B-12</td>
</tr>
<tr>
<td>Request-Type</td>
<td>B-12</td>
</tr>
<tr>
<td>Require-User-To-Be-In-Authorization-List</td>
<td>B-13</td>
</tr>
<tr>
<td>Response-Type</td>
<td>B-13</td>
</tr>
<tr>
<td>Retrace-Packet</td>
<td>B-14</td>
</tr>
<tr>
<td>Send-PEAP-URI-TLV</td>
<td>B-14</td>
</tr>
<tr>
<td>Session-Key</td>
<td>B-14</td>
</tr>
<tr>
<td>Session-Manager</td>
<td>B-14</td>
</tr>
<tr>
<td>Session-Notes</td>
<td>B-14</td>
</tr>
<tr>
<td>Session-Service</td>
<td>B-14</td>
</tr>
<tr>
<td>Set-Session-Mgr-And-Key-Upon-Lookup</td>
<td>B-15</td>
</tr>
<tr>
<td>Skip-Session-Management</td>
<td>B-15</td>
</tr>
<tr>
<td>Skip-Overriding-Username-With-LDAP-UID</td>
<td>B-15</td>
</tr>
<tr>
<td>Skip-Overriding-UserName-With-PEAP-identity</td>
<td>B-15</td>
</tr>
<tr>
<td>Source-IP-Address</td>
<td>B-15</td>
</tr>
<tr>
<td>Source-Port</td>
<td>B-15</td>
</tr>
<tr>
<td>SQL-Sequence</td>
<td>B-16</td>
</tr>
<tr>
<td>Subnet-Size-If-No-Match</td>
<td>B-16</td>
</tr>
<tr>
<td>Trace-Level</td>
<td>B-16</td>
</tr>
<tr>
<td>Unavailable-Resource</td>
<td>B-16</td>
</tr>
<tr>
<td>Unavailable-Resource-Type</td>
<td>B-16</td>
</tr>
<tr>
<td>UserDefined1</td>
<td>B-16</td>
</tr>
<tr>
<td>User-Authorization-Script</td>
<td>B-16</td>
</tr>
</tbody>
</table>
Overview

The chapter provides an overview of the RADIUS server, including connection steps, RADIUS message types, and using Cisco Prime Access Registrar (Prime Access Registrar) as a proxy server.

Prime Access Registrar is a 3GPP-compliant, 64 bit carrier-class RADIUS (Remote Authentication Dial-In User Service)/Diameter server that enables multiple dial-in Network Access Server (NAS) devices to share a common authentication, authorization, and accounting database.

Prime Access Registrar handles the following tasks:

- **Authentication**—determines the identity of users and whether they can be allowed to access the network
- **Authorization**—determines the level of network services available to authenticated users after they are connected
- **Accounting**—keeps track of each user’s network activity
- **Session and resource management**—tracks user sessions and allocates dynamic resources

Using a RADIUS server allows you to better manage the access to your network, as it allows you to store all security information in a single, centralized database instead of distributing the information around the network in many different devices. You can make changes to that single database instead of making changes to every network access server in your network.

Prime Access Registrar also allows you to manage the complex interconnections of the new network elements in order to:

- adequately manage the traffic
- perform appropriate load balancing for desired load distribution
- allow binding of different protocol interfaces corresponding to a subscriber/network element

Service providers transform their 3G and 4G wireless networks with complex services, tiered charging, converged billing, and more by introducing increasing numbers and types of Diameter-based network elements. LTE and IMS networks are the most likely to implement these new network elements—including Policy and Charging Rules Functions (PCRF), Home Subscriber Servers (HSS), Mobility Management Entities (MME), Online Charging Systems (OCS), and others. As a result, as the traffic levels grow, these wireless networks are becoming more difficult to manage and scale without the Prime Access Registrar infrastructure.
This chapter contains the following sections:

- Prime Access Registrar Hierarchy
- Prime Access Registrar Directory Structure
- Program Flow
- RADIUS Protocol
- Service and Ports Used in Prime Access Registrar

Prime Access Registrar Hierarchy

Prime Access Registrar's operation and configuration is based on a set of objects. These objects are arranged in a hierarchical structure much like the Windows 95 Registry or the UNIX directory structure. Prime Access Registrar's objects can themselves contain subobjects, just as directories can contain subdirectories. These objects include the following:

- Radius— the root of the configuration hierarchy
- UserLists—contains individual UserLists which in turn contain users
- UserGroups—contains individual UserGroups
- Users—contains individual authentication or authorization details of a user
- Clients—contains individual Clients
- Vendors—contains individual Vendors
- Scripts—contains individual Scripts
- Policies—contains a set of rules applied to an Access-Request
- Services—contains individual Services
- CommandSets—contains commands and the action to perform during Terminal Access Controller Access-Control System Plus (TACACS+) command authorization
- DeviceAccessRules—contains conditions or expressions and the applicable command sets for TACACS+ command authorization
- FastRules—provides a mechanism to easily choose the right authentication, authorization, accounting, and query service(s), drop, reject, or break flows, choose session manager or other rules required for processing a packet
- SessionManagers—contains individual Session Managers
- ResourceManagers—contains individual Resource Managers
- Profiles—contains individual Profiles
- RemoteServers—contains individual RemoteServers
- Advanced—contains Ports, Interfaces, Reply Messages, and the Attribute dictionary

This section contains the following topics:

- UserLists and Groups
- Profiles
- Scripts
- Services
- Session Management Using Resource Managers
UserLists and Groups

Prime Access Registrar lets you organize your user community through the configuration objects UserLists, users, and UserGroups.

- Use UserLists to group users by organization, such as Company A and Company B. Each list contains the actual names of the users.
- Use Users to store information about particular users, such as name, password, group membership, base profile, and so on.
- Use UserGroups to group users by function, such as PPP, Telnet, or multiprotocol users. Groups allow you to maintain common authentication and authorization requirements in one place, and have them referenced by many users.

For more information about UserLists and UserGroups, see UserLists and Groups in Chapter 4, “Cisco Prime Access Registrar Server Objects.”

Profiles

Prime Access Registrar uses Profiles that allow you to group RADIUS attributes to be included in an Access-Accept packet. These attributes include values that are appropriate for a particular user class, such as PPP or Telnet user. The user’s base profile defines the user’s attributes, which are then added to the response as part of the authorization process.

Although you can use Group or Profile objects in a similar manner, choosing whether to use one rather than the other depends on your site. If you require some choice in determining how to authorize or authenticate a user session, then creating specific profiles, and specifying a group that uses a script to choose among the profiles is more flexible. In such a situation, you might create a default group and then write a script that selects the appropriate profile based on the specific request. The benefit to this technique is each user can have a single entry, and use the appropriate profile depending on the way they log in.

For more information about Profiles, see Profiles in Chapter 4, “Cisco Prime Access Registrar Server Objects.”

Scripts

Prime Access Registrar allows you to create scripts you can execute at various points within the processing hierarchy.

- Incoming scripts—enable you to read and set the attributes of the request packet, and set or change the Environment dictionary variables. You can use the environment variables to control subsequent processing, such as specifying the use of a particular authentication service.
- Outgoing scripts—enable you to modify attributes returned in the response packet.

For more information about Scripts, see Scripts in the Chapter 4, “Cisco Prime Access Registrar Server Objects.”

Services

Prime Access Registrar uses Services to let you determine how authentication, authorization, and/or accounting are performed.
For example, to use Services for authentication:

- When you want the authentication to be performed by the Prime Access Registrar RADIUS server, you can specify the local service. In this case, you must specify a specific UserList.
- When you want the authentication performed by another server, which might run an independent application on the same or different host than your RADIUS server, you can specify either a radius, ldap, or tacacs-udp service. In this case, you must list these servers by name.

When you have specified more than one authentication service, Prime Access Registrar determines which one to use for a particular Access-Request by checking the following:

- When an incoming script has set the Environment dictionary variable Authentication-Service with the name of a Service, Prime Access Registrar uses that service.
- Otherwise, Prime Access Registrar uses the default authentication service. The default authentication service is a property of the Radius object.

Prime Access Registrar chooses the authentication service based on the variable Authentication-Service, or the default. The properties of that Service, specify many of the details of that authentication service, such as, the specific user list to use or the specific application (possibly remote) to use in the authentication process.

For more information about Services, see Services in the Chapter 4, “Cisco Prime Access Registrar Server Objects.”

### Session Management Using Resource Managers

Prime Access Registrar lets you track user sessions, and/or allocate dynamic resources to users for the lifetime of their session. You can define one or more Session Managers, and have each one manage the sessions for a particular group or company.

Session Managers use Resource Managers, which in turn manage resources of a particular type as described below.

- IP-Dynamic—manages a pool of IP addresses and allows you to dynamically allocate IP addresses from that pool
- IP-Per-NAS-Port—allows you to associate ports to specific IP addresses, and thus ensure each NAS port always gets the same IP address
- IPX-Dynamic—manages a pool of IPX network addresses
- Subnet-Dynamic—manages a pool of subnet addresses
- Group-Session-Limit—manages concurrent sessions for a group of users; that is, it keeps track of how many sessions are active and denies new sessions after the configured limit has been reached
- User-Session-Limit—manages per-user concurrent sessions; that is, it keeps track of how many sessions each user has and denies the user a new session after the configured limit has been reached
- Home-Agent—manages a pool of on-demand IP addresses
- USR-VPN—manages Virtual Private Networks (VPNs) that use USR NAS Clients
- Home-Agent-IPv6—manages a pool of on-demand IPv6 addresses
- Remote-IP-Dynamic—manages a pool of IP addresses that allows you to dynamically allocate IP addresses from a pool of addresses. It internally works with a remote ODBC database.
- Remote-User-Session-Limit—manages per-user concurrent sessions; that is, it keeps track of how many sessions each user has and denies the user a new session after the configured limit has been reached. It internally works with a remote ODBC database.
Remote-Group-Session-Limit—manages concurrent sessions for a group of users; that is, it keeps track of how many sessions are active and denies new sessions after the configured limit has been reached. It internally works with a remote ODBC database.

Session Cache—allows you to define the RADIUS attributes to store in cache.

Dynamic-DNS—manages the DNS server.

Remote-Session-Cache—allows you to define the RADIUS attributes to store in cache. It should be used with session manager of type 'remote'.

3GPP—allows you to define the attribute for 3GPP authorization.

For more information about Session Managers, see Session Managers in Chapter 4, “Cisco Prime Access Registrar Server Objects.”

If necessary, you can create a complex relationship between the Session Managers and the Resource Managers.

When you need to share a resource among Session Managers, you can create multiple Session Managers that refer to the same Resource Manager. For example, if one pool of IP addresses is shared by two departments, but each department has a separate policy about how many users can be logged in concurrently, you might create two Session Managers and three Resource Managers. One dynamic IP Resource Manager that is referenced by both Session Managers, and two concurrent session Resource Managers, one for each Session Manager.

In addition, Prime Access Registrar lets you pose queries about sessions. For example, you can query Prime Access Registrar about which session (and thus which NAS-Identifier, NAS-Port and/or User-Name) owns a particular resource, as well as query Prime Access Registrar about how many resources are allocated or how many sessions are active.

**Prime Access Registrar Directory Structure**

The installation process populates the `/opt/CSCOar` directory with the subdirectories listed in Table 1-1.

<table>
<thead>
<tr>
<th>Subdirectory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.system</td>
<td>Contains ELF,s, or binary SPARC executables that should not be run directly.</td>
</tr>
<tr>
<td>bin</td>
<td>Contains shell scripts and programs frequently used by a network administrator; programs that can be run directly.</td>
</tr>
<tr>
<td>conf</td>
<td>Contains configuration files.</td>
</tr>
<tr>
<td>data</td>
<td>Contains the <code>radius</code> directory, which contains session backing files; and the <code>db</code> directory, which contains configuration database files.</td>
</tr>
<tr>
<td>examples</td>
<td>Contains documentation, sample configuration scripts, and shared library scripts.</td>
</tr>
<tr>
<td>lib</td>
<td>Contains Prime Access Registrar software library files.</td>
</tr>
<tr>
<td>logs</td>
<td>Contains system logs and is the default directory for RADIUS accounting.</td>
</tr>
<tr>
<td>odbc</td>
<td>Contains Prime Access Registrar ODBC files.</td>
</tr>
<tr>
<td>scripts</td>
<td>Contains sample scripts that you can modify to automate configuration, and to customize your RADIUS server.</td>
</tr>
<tr>
<td>temp</td>
<td>Used for temporary storage.</td>
</tr>
</tbody>
</table>
Program Flow

When a NAS sends a request packet to Prime Access Registrar with a name and password, Prime Access Registrar performs the following actions. Table 1-2 describes the flow without regard to scripting points.

Table 1-2  From Access-Request to Access-Accept

<table>
<thead>
<tr>
<th>Prime Access Registrar Server Action</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receives an Access-Request</td>
<td>The Prime Access Registrar server receives an Access-Request packet from a NAS.</td>
</tr>
<tr>
<td>Determines whether to accept the request</td>
<td>The Prime Access Registrar server checks to see if the client’s IP address is listed in /Radius/Clients/&lt;Name&gt;/&lt;IPAddress&gt;.</td>
</tr>
<tr>
<td>Invokes the policy SelectPolicy if it exists</td>
<td>The Prime Access Registrar Policy Engine provides an interface to define and configure a policy and to apply the policy to the corresponding access-request packets.</td>
</tr>
<tr>
<td>Performs authentication and/or authorization</td>
<td>Directs the request to the appropriate service, which then performs authentication and/or authorization according to the type specified in /Radius/Services/&lt;Name&gt;/&lt;Type&gt;.</td>
</tr>
<tr>
<td>Performs session management</td>
<td>Directs the request to the appropriate Session Manager.</td>
</tr>
<tr>
<td>Performs resource management for each Resource Manager in the SessionManager</td>
<td>Directs the request to the appropriate resource manager listed in /Radius/SessionManagers/&lt;Name&gt;/&lt;ResourceManagers&gt;/&lt;Name&gt;, which then allocates or checks the resource according to the type listed in /Radius/&lt;ResourceManagers&gt;/&lt;Name&gt;/&lt;Type&gt;.</td>
</tr>
<tr>
<td>Sends an Access-Accept</td>
<td>Creates and formats the response, and sends it back to the client (NAS).</td>
</tr>
</tbody>
</table>

Prime Access Registrar supports Diameter with Extensible Authentication Protocol (EAP) functionality to enable authentication between NAS and a backend NAS Diameter authentication server. For more information, see Diameter with EAP Support, page 8-2.

Prime Access Registrar also support 3GPP compliance by implementing a set of protocols. To understand more about the 3GPP AAA server support and the call flow, see 3GPP Compliance, page 20-7.

Scripting Points

Prime Access Registrar lets you invoke scripts you can use to affect the Request, Response, or Environment dictionaries. This section contains the following topics:
- Client Scripting
- Client or NAS Scripting Points
- Authentication and/or Authorization Scripting Points

**Client Scripting**

Though Prime Access Registrar allows external code (Tcl/C/C++/Java) to be used by means of a script, custom service, policy engine, and so forth, while processing request, response, or while working with the environment dictionaries, it shall not be responsible for the scripts used and will not be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of the script.

Prime Access Registrar also allows you to define internal scripts, by which you can add, modify, or delete attributes in the request, response, and environment dictionaries for RADIUS, Diameter, and TACACS+.

**Client or NAS Scripting Points**

Table 1-3 shows the location of the scripting points within the section that determines whether to accept the request from the client or NAS. Note, the scripting points are indicated with the asterisk (*) symbol.

<table>
<thead>
<tr>
<th>Action</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receives an Access-Request.</td>
<td>The Prime Access Registrar RADIUS server receives an Access-Request packet from a NAS.</td>
</tr>
<tr>
<td>Determines whether to accept the request.</td>
<td>The client’s IP address listed in /Radius/Clients/&lt;Name&gt;/IPAddress.</td>
</tr>
<tr>
<td>*Executes the server’s incoming script.</td>
<td>A script referred to in /Radius/IncomingScript.</td>
</tr>
<tr>
<td>*Executes the vendor’s incoming script.</td>
<td>The vendor listed in /Radius/Clients/Name/Vendor, and is a script referred to in /Radius/Vendors/&lt;Name&gt;/IncomingScript.</td>
</tr>
<tr>
<td>*Executes the client’s incoming script.</td>
<td>A script referred to in /Radius/Clients/&lt;Name&gt;/IncomingScript.</td>
</tr>
<tr>
<td>Determines whether to accept requests from this specific NAS.</td>
<td>/Radius/Advanced/RequireNASsBehindProxyBeInClientList set to TRUE.</td>
</tr>
</tbody>
</table>

The NAS’s Identifier listed in /Radius/Clients/<Name>, or its NAS-IP-Address listed in /Radius/Clients/<Name>/ IPAddress.

**If the client’s IP address listed in /Radius/Clients/<Name>/IPAddress is different:**

| *Executes the vendor’s incoming script. | The vendor listed in /Radius/Clients/Name/Vendor, and is a script referred to in /Radius/Vendors/<Name>/IncomingScript. |
| *Executes the client’s incoming script. | The client listed in the previous /Radius/Clients/Name, and is a script referred to in /Radius/Clients/Name/IncomingScript. |
Authentication and/or Authorization Scripting Points

Table 1-4 shows the location of the scripting points within the section that determines whether to perform authentication and/or authorization.

Table 1-4  Authentication and Authorization Scripting Points

<table>
<thead>
<tr>
<th>Action</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Determines Service to use for authentication and/or</td>
<td>The Service name defined in the Environment dictionary variable Authentication-Service, and is the same as the Service defined in the Environment dictionary variable Authorization-Service.</td>
</tr>
<tr>
<td>authorization.</td>
<td>The Service name referred to by /Radius/DefaultAuthenticationService, and is the same as the Service defined in /Radius/DefaultAuthorizationService.</td>
</tr>
<tr>
<td>Performs authentication and/or authorization.</td>
<td>If the Services are the same, perform authentication and authorization.</td>
</tr>
<tr>
<td></td>
<td>If the Services are different, just perform authentication.</td>
</tr>
<tr>
<td>*Executes the Service’s incoming script.</td>
<td>A script referred to in /Radius/Services/&lt;Name&gt;/IncomingScript.</td>
</tr>
<tr>
<td>Performs authentication and/or authorization.</td>
<td>Based on the Service type defined in /Radius/Services/&lt;Name&gt;/&lt;Type&gt;.</td>
</tr>
<tr>
<td>*Executes the Service’s outgoing script.</td>
<td>A script referred to in /Radius/Services/&lt;Name&gt;/OutgoingScript.</td>
</tr>
<tr>
<td>Determines whether to perform authorization.</td>
<td>The Service name defined in /Radius/DefaultAuthorizationService, if different than the Authentication Service.</td>
</tr>
<tr>
<td>*Executes the Service’s incoming script.</td>
<td>A script referred to in /Radius/Services/&lt;Name&gt;/IncomingScript.</td>
</tr>
<tr>
<td>Performs authorization.</td>
<td>Checks that the Service type is defined in /Radius/Services/&lt;Name&gt;/&lt;Type&gt;.</td>
</tr>
<tr>
<td>*Executes the Service’s outgoing script.</td>
<td>A script referred to in /Radius/Services/&lt;Name&gt;/OutgoingScript.</td>
</tr>
</tbody>
</table>

Session Management

The Session Management feature requires the client (NAS or proxy) to send all RADIUS accounting requests to the Prime Access Registrar server performing session management. (The only exception is if the clients are USR/3Com Network Access Servers configured to use the USR/3Com RADIUS resource management feature.) This information is used to keep track of user sessions, and the resources allocated to those sessions.

When another accounting RADIUS server needs this accounting information, the Prime Access Registrar server performing session management might proxy it to this second server.

The count-sessions /radius all command helps to count the total sessions in Prime Access Registrar. The options are similar to the query-session command options. The query-session command displays cached attributes in addition to session details.

Table 1-5 describes how Prime Access Registrar handles session management.
Chapter 1      Overview

This section contains the following topics:

- Failover by the NAS and Session Management
- Cross Server Session and Resource Management

### Failover by the NAS and Session Management

When a Network Access Server's primary RADIUS server is performing session management, and the NAS determines the server is not responding and begins sending requests to its secondary RADIUS server, the following occurs:

- The secondary server will not know about the current active sessions that are maintained on the primary server. Any resources managed by the secondary server must be distinct from those managed by the primary server, otherwise it will be possible to have two sessions with the same resources (for example, two sessions with the same IP address).
- The primary server will miss important information that allows it to maintain a correct model of what sessions are active (because the authentication and accounting requests are being sent to the secondary server). This means when the primary server comes back online and the NAS begins using it, its knowledge of what sessions are active will be out-of-date and the resources for those sessions are allocated even if they are free to allocate to someone else.

For example, the user-session-limit resource might reject new sessions because the primary server does not know some of the users using the resource logged out while the primary server was offline. It might be necessary to release sessions manually using the `aregcmd` command `release-session`.

### Cross Server Session and Resource Management

Prime Access Registrar can manage sessions and resources across AAA Server boundaries. A session can be created by an Access-Request sent to Prime AR1, and it can be removed by an Accounting-Stop request sent to Prime AR2, as shown in Figure 1-1. This enables accurate tracking of User and Group session limits across multiple AAA Servers, and IP addresses allocated to sessions are managed in one place.
Chapter 1      Overview

Program Flow

All resources that must be shared across multiple front line Prime Access Registrars are configured in the Central Resource Prime Access Registrar. Resources that are not shared can still be configured at each front line Prime Access Registrar.

When the front line Prime Access Registrar receives the access-request, it does the regular AA processing. If the packet is not rejected and a Central Resource Prime Access Registrar is also configured, the front line Prime Access Registrar will proxy the packet to the configured Central Resource Prime Access Registrar. If the Central Resource Prime Access Registrar returns the requested resources, the process continues to the local session management (if local session manager is configured) for allocating any local resources. If the Central Resource Prime Access Registrar cannot allocate the requested resource, the packet is rejected.

When the Accounting-Stop packet arrives at the frontline Prime Access Registrar, Prime Access Registrar does the regular accounting processing. Then, if the front line Prime Access Registrar is configured to use Central Resource Prime Access Registrar, a proxy packet will be sent to Central Resource Prime Access Registrar for it to release all the allocated resources for this session. After that, any locally allocated resources are released by the local session manager.

Session-Service Service Step and Radius-Session Service

A new Service step has been added in the processing of Access-Request and Accounting packets. This is an additional step after the AA processing for Access packet or Accounting processing for Accounting packet, but before the local session management processing. The Session-Service should have a service type of radius-session.

An environment variable Session-Service is introduced to determine the Session-Service dynamically. You can use a script or the rule engine to set the Session-Service environment variable.

Configure Front Line Cisco Prime Access Registrar

To use a Central Resource server, the DefaultSessionService property must be set or the Session-Service environment variable must be set through a script or the rule engine. The value in the Session-Service variable overrides the DefaultSessionService.

The configuration parameters for a Session-Service service type are the same as those for configuring a radius service type for proxy, except the service type is radius-session.

The configuration for a Session-Service Remote Server is the same as configuring a proxy server.

[ //localhost/Radius ]
Name = Radius
Description =
Version = 7.0
IncomingScript =
OutgoingScript =
DefaultAuthenticationService = local-users

1. The proxy packet is actually a resource allocation request, not an Access Request.
DefaultAuthorizationService = local-users  
DefaultAccountingService = local-file  
DefaultSessionService = Remote-Session-Service  
DefaultSessionManager = session-mgr-1  

[ //localhost/Radius/Services ]  
Remote-Session-Service/  
    Name = Remote-Session-Service  
    Description =  
    Type = radius-session  
    IncomingScript =  
    OutgoingScript =  
    OutagePolicy = RejectAll  
    OutageScript =  
    MultipleServersPolicy = Failover  
RemoteServers/  
    1. central-server  

[ //localhost/Radius/RemoteServers ]  
central-server/  
    Name = central-server  
    Description =  
    Protocol = RADIUS  
    IPAddress = 209.165.200.224  
    Port = 1812  
    ReactivateTimerInterval = 300000  
    SharedSecret = secret  
    Vendor =  
    IncomingScript =  
    OutgoingScript =  
    MaxTries = 3  
    InitialTimeout = 2000  
    AccountingPort = 1813

Configure Central Prime Access Registrar

Resources at the Central Resource server are configured the same way as local resources are configured. These resources are local resources from the Central Resource server’s point of view.

Script Processing Hierarchy

For request packets, the script processing order is from the most general to the most specific. For response packets, the processing order is from the most specific to the most general.

Table 1-6, Table 1-7, and Table 1-8 show the overall processing order and flow: (1-6) Incoming Scripts, (7-11) Authentication/Authorization Scripts, and (12-17) Outgoing Scripts.

Note

The client and the NAS can be the same entity, except when the immediate client is acting as a proxy for the actual NAS.
Prime Access Registrar is based on a client/server model, which supports AAA (authentication, authorization, and accounting). The client is the Network Access Server (NAS) and the server is Prime Access Registrar. The client passes user information on to the RADIUS server and acts on the response it receives. The server, on the other hand, is responsible for receiving user access requests, authenticating and authorizing users, and returning all of the necessary configuration information the client can then pass on to the user.

The protocol is a simple packet exchange in which the NAS sends a request packet to the Prime Access Registrar with a name and a password. Prime Access Registrar looks up the name and password to verify it is correct, determines for which dynamic resources the user is authorized, then returns an accept packet that contains configuration information for the user session (Figure 1-2).
Prime Access Registrar can also reject the packet if it needs to deny network access to the user. Or, Prime Access Registrar can issue a challenge that the NAS sends to the user, who then creates the proper response and returns it to the NAS, which forwards the challenge response to Prime Access Registrar in a second request packet.

In order to ensure network security, the client and server use a *shared secret*, which is a string they both know, but which is never sent over the network. User passwords are also encrypted between the client and the server to protect the network from unauthorized access.

This section contains the following topics:

- **Steps to Connection**
- **Types of RADIUS Messages**
- **Proxy Servers**

## Steps to Connection

Three participants exist in this interaction: the user, the NAS, and the RADIUS server.

### Setting Up the Connection

To describe the receipt of an access request through the sending of an access response:

- **Step 1** The user, at a remote location such as a branch office or at home, dials into the NAS, and supplies a name and password.
- **Step 2** The NAS picks up the call and begins negotiating the session.
  - a. The NAS receives the name and password.
  - b. The NAS formats this information into an Access-Request packet.
  - c. The NAS sends the packet on to the Prime Access Registrar server.
- **Step 3** The Prime Access Registrar server determines what hardware sent the request (NAS) and parses the packet.
  - a. It sets up the Request dictionary based on the packet information.
  - b. It runs any incoming scripts, which are user-written extensions to Prime Access Registrar. An incoming script can examine and change the attributes of the request packet or the environment variables, which can affect subsequent processing.
c. Based on the scripts or the defaults, it chooses a service to authenticate and/or authorize the user.

**Step 4** Prime Access Registrar's authentication service verifies the username and password is in its database. Or, Prime Access Registrar delegates the authentication (as a proxy) to another RADIUS server, an LDAP, or TACACS server.

**Step 5** Prime Access Registrar's authorization service creates the response with the appropriate attributes for the user's session and puts it in the Response dictionary.

**Step 6** If you are using Prime Access Registrar session management at your site, the Session Manager calls the appropriate Resource Managers that allocate dynamic resources for this session.

**Step 7** Prime Access Registrar runs any outgoing scripts to change the attributes of the response packet.

**Step 8** Prime Access Registrar formats the response based on the Response dictionary and sends it back to the client (NAS).

**Step 9** The NAS receives the response and communicates with the user, which might include sending the user an IP address to indicate the connection has been successfully established.

---

**Types of RADIUS Messages**

The client/server packet exchange consists primarily of the following types of RADIUS messages:

- Access-Request—sent by the client (NAS) requesting access
- Access-Reject—sent by the RADIUS server rejecting access
- Access-Accept—sent by the RADIUS server allowing access
- Access-Challenge—sent by the RADIUS server requesting more information in order to allow access. The NAS, after communicating with the user, responds with another Access-Request.

When you use RADIUS accounting, the client and server can also exchange the following two types of messages:

- Accounting-Request—sent by the client (NAS) requesting accounting
- Accounting-Response—sent by the RADIUS server acknowledging accounting

This section contains the following topics:

- Packet Contents
- The Attribute Dictionary

---

**Packet Contents**

The information in each RADIUS message is encapsulated in a UDP (User Datagram Protocol) data packet. A packet is a block of data in a standard format for transmission. It is accompanied by other information, such as the origin and destination of the data.

*Table 1-9* lists a description of the five fields in each message packet.
The Attribute Dictionary

The Attribute Dictionary contains a list of preconfigured authentication, authorization, and accounting attributes that can be part of a client’s or user’s configuration. The dictionary entries translate an attribute into a value Prime Access Registrar uses to parse incoming requests and generate responses. Attributes have a human-readable name and an enumerated equivalent from 1-255.

Sixty-three standard attributes exist, which are defined in RFC 2138 and 2139. There are also additional vendor-specific attributes that depend on the particular NAS you are using.

Some sample attributes include:
- User-Name—the name of the user
- User-Password—the user’s password
- NAS-IP-Address—the IP address of the NAS
- NAS-Port—the NAS port the user is dialed into
- Framed-Protocol—such as SLIP or PPP
- Framed-IP-Address—the IP address the client uses for the session
- Filter-ID—vendor-specific; identifies a set of filters configured in the NAS
- Callback-Number—the actual callback number.

Proxy Servers

Any one or all of the RADIUS server’s three functions: authentication, authorization, or accounting can be subcontracted to another RADIUS server. Prime Access Registrar then becomes a proxy server. Proxying to other servers enables you to delegate some of the RADIUS server’s functions to other servers.
You could use Prime Access Registrar to “proxy” to an LDAP server for access to directory information about users in order to authenticate them. Figure 1-3 shows user joe initiating a request, the Prime Access Registrar server proxying the authentication to the LDAP server, and then performing the authorization and accounting processing in order to enable joe to log in.

Figure 1-3 Proxying to an LDAP Server for Authentication

Service and Ports Used in Prime Access Registrar

Secure Shell Service

SSH Daemon (SSHD) is the daemon program which is used for ssh(1). It provides secure shell encrypted communications between two hosts over network.

In case of Prime Access Registrar, SSH is used to connect to Prime Access Registrar server and configure Prime Access Registrar using CLI.

Ports

The following table lists the port numbers that are used for various services in Prime Access Registrar for AAA.
## Table 1-10 Ports Used in Prime Access Registrar

<table>
<thead>
<tr>
<th>Names</th>
<th>Description</th>
<th>Port Numbers</th>
<th>Service of the Ports</th>
<th>Access from Network Node</th>
<th>Configuration Setting</th>
<th>Protocol Name and Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR AAA Service</td>
<td>The RADIUS packet listener uses these ports by default.</td>
<td>1812-udp</td>
<td>RADIUS AA</td>
<td>Network Access Server</td>
<td>You can change the default or define new RADIUS port numbers under /Radius/Advanced/Ports in the CLI and Configuration &gt; Advanced &gt; Ports in the GUI.</td>
<td>RADIUS AA (Authentication, and Authorization) service.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1813-udp</td>
<td>RADIUS Accounting</td>
<td>Network Access Server</td>
<td>You can change the default or define new RADIUS port numbers under /Radius/Advanced/Ports in the CLI and Configuration &gt; Advanced &gt; Ports in the GUI.</td>
<td>RADIUS Accounting service. Refer to RFC 6733 for more information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3799/udp</td>
<td>RADIUS Dynamic Authorization (CoA/PoD)</td>
<td>Network Access Server</td>
<td>N/A</td>
<td>RADIUS Dynamic authorization which is used with (CoA/PoD) packet types.</td>
</tr>
<tr>
<td>AR AAA Service</td>
<td>The TACACS+ packet listener uses this port by default.</td>
<td>49/tcp</td>
<td>TACACS+</td>
<td>Network Access Server</td>
<td>You can change the default or define new RADIUS port numbers under /Radius/Advanced/Ports in the CLI and Configuration &gt; Advanced &gt; Ports in the GUI.</td>
<td>TACACS+ based on AAA service (Authentication, Authorization, and Accounting). Refer to RFC 1491 for more information.</td>
</tr>
</tbody>
</table>
Table 1-10  Ports Used in Prime Access Registrar (continued)

<table>
<thead>
<tr>
<th>Names</th>
<th>Description</th>
<th>Port Numbers</th>
<th>Service of the Ports</th>
<th>Access from Network Node</th>
<th>Configuration Setting</th>
<th>Protocol Name and Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR AAA Service</td>
<td>The DIAMETER packet listener uses these ports by default.</td>
<td>3868/tcp</td>
<td>DIAMETER</td>
<td>Network Access Server</td>
<td>You can enable or disable this service in Radius/Advanced/Diameter/IsDiameterEnabled.</td>
<td>DIAMETER AA Service (Authentication, and Authorization) by tcp protocol. Refer to RFC 4005 for more information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3868/sctp</td>
<td>DIAMETER</td>
<td>Network Access Server</td>
<td>You can enable or disable this service in Radius/Advanced/Diameter/IsDiameterEnabled.</td>
<td>DIAMETER AA Service (Authentication, and Authorization) by SCTP protocol.</td>
</tr>
<tr>
<td>AR MCD Server</td>
<td>MCD is used to store Prime Access Registrar configuration.</td>
<td>2786/tcp</td>
<td>MCD database Server</td>
<td>This service can be accessed from local host by Prime Access Registrar radius and server agent process.</td>
<td>N/A</td>
<td>Proprietary IPC mechanism.</td>
</tr>
<tr>
<td>AR Server Agent</td>
<td>AR Server Agent is used to log all the activities of Prime Access Registrar processes.</td>
<td>2785/tcp</td>
<td>Internal IPC mechanism</td>
<td>This service can be accessed from local host by Prime Access Registrar radius and server agent process.</td>
<td>N/A</td>
<td>Proprietary IPC mechanism.</td>
</tr>
</tbody>
</table>
## Table 1-10 Ports Used in Prime Access Registrar (continued)

<table>
<thead>
<tr>
<th>Names</th>
<th>Description</th>
<th>Port Numbers</th>
<th>Service of the Ports</th>
<th>Access from Network Node</th>
<th>Configuration Setting</th>
<th>Protocol Name and Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR GUI Service</td>
<td>Prime Access Registrar GUI processes use these ports by default.</td>
<td>8080/tcp</td>
<td>AR HTTP service</td>
<td>This service is accessible from any end user desktop browser using http protocol.</td>
<td>You can change the default port numbers in editing the server.xml file.</td>
<td>Standard HTTP protocol</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8443/tcp</td>
<td>AR HTTPS service</td>
<td>This service is accessible from any end user desktop browser using https protocol.</td>
<td>You can change the default port numbers in editing the server.xml file.</td>
<td>Standard HTTPS protocol</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8005/tcp</td>
<td>Internally used by Apache Tomcat container</td>
<td>Local host</td>
<td>You can change the default port numbers in editing the server.xml file.</td>
<td>To shutdown Tomcat JVM service instance.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8009/tcp</td>
<td>Apache Tomcat container AJP 1.3 Connector</td>
<td>Local host</td>
<td>You can change the default port numbers in editing the server.xml file.</td>
<td>Apache JServ protocol. AJP 1.3 Connector.</td>
</tr>
<tr>
<td>SNMP Master Agent</td>
<td>SNMP Packet listener supports these ports by default.</td>
<td>161/udp</td>
<td>Simple Net Management Protocol</td>
<td>This service is accessible from any network management host.</td>
<td>Refer to net-snmp documentation for more information.</td>
<td>SNMP MIBs server</td>
</tr>
<tr>
<td></td>
<td></td>
<td>162/udp</td>
<td>Traps for SNMP</td>
<td>This service is accessible to any SNMP trap client when you want to use net-snmp snmptrap daemon as a SNMP trap server.</td>
<td>Refer to Configuring Traps for more information.</td>
<td>SNMP trap server</td>
</tr>
<tr>
<td>Names</td>
<td>Description</td>
<td>Port Numbers</td>
<td>Service of the Ports</td>
<td>Access from Network Node</td>
<td>Configuration Setting</td>
<td>Protocol Name and Reference</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------</td>
<td>---------------------------------------</td>
<td>------------------------------------------------------------------------------------------</td>
<td>-----------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>CPAR SIGTRAN Stack (radius)</td>
<td>Listen on these ports for internal configuration from stack manager events</td>
<td>9041/TCP</td>
<td>Stack Manager Configuration/Event Listener</td>
<td>This service can be accessed from local host by Prime Access Registrar – Radius Process.</td>
<td>N/A</td>
<td>CPAR Specific IPC Protocol implementation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9041/UDP</td>
<td>Stack Manager Configuration/Event Listener</td>
<td>This service can be accessed from local host by Prime Access Registrar – Radius Process.</td>
<td>N/A</td>
<td>CPAR Specific IPC Protocol implementation</td>
</tr>
<tr>
<td>CPAR SIGTRAN stack manager(m3ua-stackmgr)</td>
<td>Configure stack and receive configuration from m3ua-cliclient</td>
<td>9100/TCP</td>
<td>SIGTRAN Stack Manager</td>
<td>This service can be accessed from local host by Prime Access Registrar – Radius Process and m3ua-cliclient Process.</td>
<td>N/A</td>
<td>CPAR Specific IPC Protocol implementation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9100/UDP</td>
<td>SIGTRAN Stack Manager</td>
<td>This service can be accessed from local host by Prime Access Registrar – Radius Process and m3ua-cliclient Process.</td>
<td>N/A</td>
<td>CPAR Specific IPC Protocol implementation</td>
</tr>
</tbody>
</table>

1. If an error occurs while starting the Diameter SCTP interface, add `install sctp /bin/true` to `/etc/modprobe.conf`. Then, configure port 3868 with Type Diameter-TCP using aregcmd in `/Radius/Advanced/Ports`. 
Related Documentation

The following documentation is available for Prime Access Registrar 7.0:

- Cisco Prime Access Registrar 7.0 User Guide (this guide)
- Cisco Prime Access Registrar 7.0 Release Notes
- Cisco Prime Access Registrar 7.0 Installation and Configuration Guide
- Open Source Used in Cisco Prime Access Registrar 7.0

Note
We sometimes update the documentation after original publication. Therefore, you should also review the documentation on Cisco.com for any updates.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, using the Cisco Bug Search Tool (BST), submitting a service request, and gathering additional information, see What’s New in Cisco Product Documentation.

To receive new and revised Cisco technical content directly to your desktop, you can subscribe to the What’s New in Cisco Product Documentation RSS feed. The RSS feeds are a free service.
Using the aregcmd Commands

This chapter describes how to use each of the aregcmd commands. The Cisco Prime Access Registrar aregcmd command is a command-line based configuration tool. It allows you to set any Cisco Prime Access Registrar (Prime Access Registrar) configurable option, as well as, start and stop the server and check statistics.

This chapter contains the following sections:

- General Command Syntax
- aregcmd Commands
- aregcmd Command Logging
- aregcmd Command Line Editing
- aregcmd Error Codes

General Command Syntax

Prime Access Registrar stores its configuration information in a hierarchy. Using the aregcmd command cd (change directory), you can move through this information in the same manner as you would through any hierarchical file system. Or you can supply full pathnames to these commands to affect another part of the hierarchy, and thus avoid explicitly using the cd command to change to that part of the tree.

- aregcmd command parsing is case insensitive, which means you can use upper or lowercase letters to designate elements. In addition, when you reference existing elements in the configuration, you need only specify enough of the element’s name to distinguish it from the other elements at that level. For example, instead of entering cd Administrators, you can enter cd ad when no other element at the current level begins with ad.

- aregcmd command parsing is command-line order dependent; that is, the arguments are interpreted based on their position on the command line. To indicate an empty string as a place holder on the command line, use either single (’) or double quotes (“”). In addition, when you use any arguments that contain spaces, you must quote the arguments. For example, when you use the argument, “Local Users,” you must enclose the phrase in quotes.

The aregcmd command can contain a maximum of 255 characters when specifying a parameter and 511 characters for the entire command.

The aregcmd command syntax is:

```
aregcmd [-C <clusternamex>] [-N <adminname>] [-P <adminpassword>] [-V]
```

- -C—Specifies the name of the cluster to log into by default
General Command Syntax

- -N—Specifies the name of the administrator
- -P—Specifies the password
- -V—Specifies view-only mode
- -f—Specifies a file that can contain a series of commands
- -I—Specifies a directory where the Prime Access Registrar license file is stored and returns information about licensed components
- -n—Turns off prefix mode
- -p—Specifies prefix mode
- -q—Turns off verbose mode
- -v—Specifies verbose mode

Note

The verbose (-v) and prefix (-p) modes are on by default when you run aregcmd interactively (for example, not entered on the command line or not running commands from a script file). Otherwise, verbose and prefix modes are off.

When you include a command (with the appropriate arguments) on the command line, aregcmd runs only that one command and saves any changes.

This section contains the following topics:

- View-Only Administrator Mode
- Configuration Objects
- aregcmd Command Performance

View-Only Administrator Mode

Previous releases of Prime Access Registrar provided only super-user administrative access. If you were able to log into aregcmd, you could do anything to the system, including starting and stopping the system and changing the configuration. Prime Access Registrar provides view-only administrative access. View-only access restricts an administrator to only being able to observe the system and prevents that user from making changes.

View-only access can be encountered in three ways:

- Specific administrators can be restricted to view-only access whenever they log in.
- Administrators not restricted to view-only access can choose to start aregcmd in a view-only mode. This might be used when an administrator wants to ensure that he or she does not make any changes.
- When an administrator who is not view-only logs in to a slave server, they will be unable to make changes to any parts of the configuration other than /Radius/Replication, /Radius/Advanced/Ports, /Radius/Advanced/Interfaces or the properties in /Radius/Advanced. This is because the rest of the configuration is replicated from the master server and changes directly to the slave will cause problems.

Note

When a user logs in, the system determines whether a user's session is view-only or not. If the configuration is changed after a user has logged in, that change does not take effect until the affected user logs out and logs back in.
ViewOnly Property

The ViewOnly property has been added to the Administrators configuration. The default setting for the ViewOnly property is FALSE. The following shows the default setting for the admin user:

```
cd /Administrators/admin

[ //localhost/Administrators/admin ]
Name = admin
Description =
Password = <encrypted>
ViewOnly = FALSE
```

You can designate specific administrators to be view-only administrators by setting the new ViewOnly property to TRUE.

- If that property is set to TRUE, any time the administrator logs in to aregcmd the session will be in view-only mode.
- If set to FALSE, when the administrator logs in to a master server, the session will be full super-user capability.

If the administrator logs in to a slave, they only part of the configuration they will be able to modify is that part under `/Radius/Replication`, `/Radius/Advanced/Ports`, `/Radius/Advanced/Interfaces` or the properties in `/Radius/Advanced`.

When in a view-only session, the following commands will cause an error: `add`, `delete`, `set`, `unset`, `insert`, `validate`, `save`, `start`, `stop`, `reload`, `reset-stats`, `release-sessions`, and `trace`. The following error message will be displayed:

```
316 Command failed: session is ViewOnly
```

When in a slave server session, the following commands will cause an error when the object or property being operated on is not under `/Radius/Replication`, `/Radius/Advanced/Ports`, `/Radius/Advanced/Interfaces` or the properties in `/Radius/Advanced`: `add`, `delete`, `set`, `unset`, and `insert`. The following error message will be displayed:

```
317 Command failed: session is ViewOnly
```

Configuration Objects

The Prime Access Registrar aregcmd command lets you manipulate configuration objects, that define properties or the behavior of the RADIUS server, such as valid administrators and types of services. For descriptions of those objects, see Chapter 4, “Cisco Prime Access Registrar Server Objects.”

aregcmd Command Performance

You can impact aregcmd command performance and server response time by having Prime Access Registrar userlists that contain more than 10,000 users. Prime Access Registrar userlists were not designed to contain 10,000 users in any one list.

If you must provide service for groups greater than 10000 users, we recommend that you use an external data store such as an LDAP directory or an Oracle database. If you are unable to use an external data store, create multiple userlists instead, keeping each userlist under 10,000 users.
Multiple userlists require multiple services (one for each userlist), because a service cannot reference more than one userlist. The multiple services can then be combined using the Service Grouping feature with ResultRule, OR, as follows:

```
[ //localhost/Radius/ Services/ GroupService ]
Name = GroupService
Description =
Type = group
IncomingScript- =
OutgoingScript- =
ResultRule = OR
GroupServices/
  1. UserService1
  2. UserService2
  3. UserService3
```

**RPC Bind Services**

The Prime Access Registrar server and the aregcmd CLI requires RPC services to be running before the server is started. If the RPC services are stopped, you must restart RPC services, then restart the Prime Access Registrar server.

Use the following commands to restart RPC services:

```
arserver stop
/etc/init.d/rpc start
arserver start
```

If RPC services are not running, the following message is displayed when you attempt to start aregcmd:

```
Login to aregcmd fails with the message: 400 Login failed
```

**aregcmd Commands**

This section contains the complete list of aregcmd commands. You can use them on the command line or insert them into scripts. The commands are listed alphabetically.

This section contains the following topics:

- add
- cd
- delete
- exit
- filter
- find
- help
- insert
- login
- logout
- ls
- next
- prev
- pwd
- query-sessions
- quit
- release-sessions
- reload
- reset-stats
- save
- set
- start
- stats
- status
- stop
- tacacs-stats
- tacacs-reset-stats
- dia-stats
- trace
- trace-file-count
- unset
- validate

**add**

Use the `aregcmd` command **add** to create new elements in the configuration. The **add** command is context sensitive, which means the type of element added is determined by the current context, or the path specified as the first parameter. The **add** command has one required argument; the name of the element you wish to add. You can also provide other parameters, or you can supply this information after **aregcmd** has added the new element. The optional second argument is a description of the element.

The syntax is:

```
add [<path>/]<name> [...]
```

**cd**

Use the **aregcmd** command **cd** to change the working context, or level in the configuration hierarchy. When you use the **cd** command without any parameters, it returns you to the root of the tree. When you use the optional path argument, you can specify a new context. To change to a higher level in the tree hierarchy, use the “..” syntax (as you would in a UNIX file system). When you change to a new context, **aregcmd** displays the contents of the new location, when you are using the command in interactive mode, or if verbose mode is on.
The syntax is:
```
cd [<path>]
```

**delete**

Use the `aregcmd` command **delete** to remove an element from the configuration hierarchy. You cannot remove properties on an element; you can only remove entire elements. The **delete** command is recursive; that is, it will remove any subelements contained within an element being removed. When the element is in the current context, you need only provide the name of the element to be deleted. You can optionally provide a complete path to an element elsewhere in the configuration hierarchy.

The syntax is:
```
delete [<path>/]<name>
```

**exit**

Use the `aregcmd` command **exit** to terminate your `aregcmd` session. If you have any unsaved modifications, Prime Access Registrar asks if you want to save them before exiting. Any modifications you don’t choose to save are lost.

The syntax is:
```
exit
```

**filter**

Use the `aregcmd` command **filter** to display a selected view of a list. You can use the **filter** command to present only the elements of a list that have properties equal to the value you specify. You can also use the **filter** command to restore the view of the list after it has been filtered.

When using the **filter** command, you must provide a property name and a value, and you can optionally provide the path to the list. Prime Access Registrar displays a list with only those elements that have a value equal to the specified value. When you want to filter the current context, you can omit the path argument.

The **filter** command is **sticky**, in that, after you have filtered a list, you must explicitly unfilter it before you can view the complete list again. To restore the unrestricted view of the list, use the **filter** command and specify the string **all**. To restore the list in current context, you can omit the pathname.

The syntax is:
```
filter [<path>] <property> <value>
```
or
```
filter [<path>] all
```

**find**

Use the `aregcmd` command **find** to locate a specific item in a list. The **find** command takes one required argument, which is a full or partial pathname. After you use the command, Prime Access Registrar displays a page beginning with the entry that most closely matches the pathname you provided.
The syntax is:

```
find <path>
```

**help**

Use the `aregcmd` command `help` (with no argument specified) to display a brief overview of the command syntax. When you specify the name of a command, Prime Access Registrar displays help for only that command.

The syntax is:

```
help [command]
```

**insert**

Use the `aregcmd` command `insert` to add an item anywhere in ordered list. The required parameters are the numeric index of the position in the list in which you want to insert the new item, and the item value. When the list to which you are adding is not the current context, you can specify the complete path to the position in the list by prepending the path for the list to the numeric index. After the new value has been inserted into the list, Prime Access Registrar appropriately renumbers the list.

The syntax is:

```
insert [path]/<index> <value>
```

This command applies to lists of servers by index and the Resource Managers list in Session Managers.

**login**

Use the `aregcmd` command `login` to connect to a cluster, which contains the RADIUS server and definition of the authorized administrators. When you do not specify the cluster, admin name, and password, `aregcmd` prompts you for them.

When you are currently logged in to a cluster, the `login` command allows you to connect to another cluster. When you have changes in the current cluster that you have not saved, `aregcmd` asks if you want to save them before logging into another cluster. Any changes you do not save are lost.

After you successfully log in, and if the server is running, Prime Access Registrar displays the cluster server's health. Note, to log into a cluster, the Prime Access Registrar Server Agent for that cluster must be running.

The syntax is:

```
login [cluster] [name] [password]
```

**logout**

Use the `aregcmd` command `logout` to log out of the current cluster. After you log out, you have to log into make any modifications to the configuration hierarchy, or to manage the server(s). When you have any unsaved modifications, Prime Access Registrar asks if you want to save them before logging out. Any modifications you do not choose to save are lost.
The syntax is:

logout

**ls**

Use the `aregcmd` command `ls` to list the contents of a level in the configuration hierarchy. This command works much like the UNIX `ls` command. When you use it without any parameters, it lists the items in the current context. When you specify a path, it lists the elements found in that context. When you use the `-R` argument, it recursively lists all of the elements in and below the specified (or current) context.

For similar commands, refer to the `next` and `prev` commands.

The syntax is:

```
ls [-R] [<path>]
```

**next**

Use the `aregcmd` command `next` to review the remaining pages produced from the `ls` command. Every time you use the `cd` command, it automatically invokes the `ls` command to display the contents of the location. When the output from the `ls` command is more than one page (a page is about 24 lines) in length, Prime Access Registrar displays only the first page.

The `next` command takes an optional path and count. The path specifies the context in which you wish to see the next page and the count specifies the number of lines you wish to see. When you use the `next` command without the path, Prime Access Registrar uses the current context. When you do not specify a count, Prime Access Registrar uses the last count value you used with the `next` or `prev` command. If you never specify a count, Prime Access Registrar uses the default value, which is 20.

Note, the current page for a context is sticky. This means, for example, when you use the `next` command to view entries 20 through 30, until you use the `next` or `prev` command on the same context, you will continue to see these entries even if you use the `cd` command to change to a different context, then return to the original.

The syntax is:

```
next [<path>] [<count>]
```

**prev**

Use the `aregcmd` command `prev` to page backwards through the output of the `ls` command. It behaves much like the `next` command, in that it takes an optional path identifying a context to display and a count parameter indicating how many lines to display.

The syntax is:

```
prev [<path>] [<count>]
```
**pwd**

Use the `aregcmd` command `pwd` to display the absolute pathname of the current context (level in the configuration hierarchy).

The syntax is:

```
pwd
```

**query-sessions**

Use the `aregcmd` command `query-sessions` to query the server about the currently active user sessions. You can request information about all of the active sessions or just those sessions that match the type you specify.

The syntax is:

```
query-sessions <path> [all]
```

or

```
query-sessions <path> with-type <value> [send-CoA [with-profile <profile name>]]
```

or

```
query-sessions <path> with-Attribute <name> <value> [send-CoA [with-profile <profile name>]]
```

Where `<path>` is the path to the server, Session Manager, or Resource Manager to query and `with-type` is one of the following: `with-NAS`, `with-User`, `with-IP-Address`, `with-IPX-Network`, `with-USR-VPN`, `with-Key`, `with-ID` or `with-Age`. The optional `[with-profile <profile name>]` parameter indicates a profile name as configured in `/Radius/Profiles`.

The `query-sessions` command with an optional `[send-CoA]` at the end causes the Prime Access Registrar server to send a Change of Authorization (CoA) request to the client. The CoA request includes the CoA attributes configured for the client. When the optional profile name is also included in the command, the Prime Access Registrar server includes the attribute-value (AV) pairs from the corresponding profile in `/Radius/Profiles` in the CoA request.

**quit**

Use the `aregcmd` command `quit` to terminate your `aregcmd` session. You can use it interchangeably with the `exit` command.

The syntax is:

```
quit
```

When you quit the `aregcmd` command, if you have made changes, the Prime Access Registrar server asks if you want to save the changes. Any unsaved changes are lost.

**release-sessions**

Use the `aregcmd` command `release-sessions` to request the server to release one or more currently active user sessions. This command might be useful, for example, in the case where you have taken a NAS offline, however, the server believes user sessions for that NAS are still active.
The syntax is one of:

```
release-sessions <path> all
```

or

```
release-sessions <path> with- <type> <value> [send-pod] [send-notification]
```

or

```
release-sessions <path> with-Attribute <name> <value> [send-pod] [send-notification]
```

Where `<path>` is the path to the server, Session Manager, or Resource Manager to query and `with-<type>` is one of the following: `with-NAS`, `with-User`, `with-IP-Address`, `with-IPX-Network`, `with-USR-VPN`, `with-Key`, or `with-ID`.

The optional `[send-pod <send notification>]` parameter sends the disconnect packet to the NAS to clear sessions and an Accounting-Stop notification to the client listed in the session record.

The optional `with-Attribute` parameter enables release a session based on a specific attribute and value.

---

### reload

Use the `aregcmd` command `reload` to stop the server (when it is running), and then immediately start the server, forcing it to reread its configuration information. When you have modified the configuration hierarchy, Prime Access Registrar asks you if you want to save your changes before restarting the server. You must save your changes in order for the reloaded server to be able to use them.

The syntax is:

```
reload
```

---

### reset-stats

Use the `aregcmd` command `reset-stats` to reset all server statistics displayed with the `stats` command. The `reset-stats` command also resets SNMP counters.

The `reset-stats` command provides a way of resetting the server statistics without having to reload or restart the server.

The syntax is:

```
reset-stats
```

---

### save

Use the `aregcmd` command `save` to validate the changes you made and commit them to the configuration database, if no errors are found.

**Note** Using the `save` command does not automatically update the running server. To update the server, you must use the `reload` command.

The syntax is:

```
save
```

Table 2-1 lists the RADIUS server objects and the effect of Dynamic Updates upon them.
set

Use the `aregcmd` command `set` to provide values for properties on existing configuration elements. You only need to provide the `set` command with the name of the property you wish to set (or just enough of the name to distinguish it from other properties) and the new value for that property. It also applies to the `Profiles` attribute list, the `Rules` attributes list, the enumeration list in the `Attribute` dictionary, and the `LDAPtoRadiusMappings` and `LDAPtoEnvironmentMappings` mappings.

The `set` command can also be used to order servers in a list. To specify a new position in a list for a server, use the `set` command and provide the numeric position of the server and the server’s name.

The syntax is:

```
set [<path>/]<property> <value>
```

When the list is a list of servers by index, the syntax is:

```
set [<path>/]<index> <server name>
```

**Note** If the index is already in use, the old server name will be replaced by the new server name.
To remove a value from a property (make a property equal to NULL), use a pair of single or double quotes as the value, as shown below:

```
set <property> ""
```

When you need to set an attribute to a value that includes a space, you must double-quote the value, as in the following:

```
set Framed-Route "192.168.1.0/24 192.168.1.1"
```

**start**

Use the `aregcmd` command `start` to enable the server to handle requests. When the configuration hierarchy has been modified, Prime Access Registrar asks you if you want to save the changes before starting the server.

The syntax is:

```
start
```

**stats**

Use the `aregcmd` command `stats` to provide statistical information on the specified server. You can only issue this command when the server is running.

Note that `aregcmd` supports the `PAGER` environment variable. When the `aregcmd stats` command is used and the `PAGER` environment variable is set, the `stats` command output is displayed using the program specified by the `PAGER` environment variable.

The syntax is:

```
stats
```

The following is an example of the statistical information provided for remote server after you issue the `stats` command:

```
RemoteServer statistics for: ServerA, 209.165.201.1, port 1812
active = TRUE
maxTries = 3
RTTAverage = 438ms
RTTDerivation = 585ms
TimeoutPenalty = 0ms
totalRequestsPending = 0
totalRequestsSent = 14
totalRequestsOutstanding = 0
totalRequestsTimedOut = 0
localRequestsAcknowledged = 14
localResponsesDroppedForNotInCache = 0
localResponsesDroppedForSignatureMismatch = 0
localRequestsDroppedAfterMaxTries = 0
lastRequestTime = Mon Feb 18 17:19:46 2013
lastAcceptTime = Mon Feb 18 17:18:11 2013
```
Table 2-2 lists the statistics displayed for the remote server by the stats command and the meaning of the values.

<table>
<thead>
<tr>
<th>Stats Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>RemoteServer statistics for:</td>
<td>Provides server’s type, name, IP address, and port used</td>
</tr>
<tr>
<td>active</td>
<td>Indicates whether the server was active (not in a down state)</td>
</tr>
<tr>
<td>maxTries</td>
<td>Number of retry attempts to be made by the RemoteServer Object based on the RemoteServer’s maxTries property setting</td>
</tr>
<tr>
<td>RTTAverage</td>
<td>Average round trip time since the last server restart</td>
</tr>
<tr>
<td>RTTDeviation</td>
<td>Indicates a standard deviation of the RTTAverage</td>
</tr>
<tr>
<td>TimeoutPenalty</td>
<td>Indicates any change made to the initial timeout default value</td>
</tr>
<tr>
<td>totalRequestsPending</td>
<td>Number of requests currently queued</td>
</tr>
<tr>
<td>totalRequestsSent</td>
<td>Number of requests sent since the last server restart</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> totalRequestsSent should equal the sum of totalRequestsPending and totalRequestsAcknowledged.</td>
</tr>
<tr>
<td>totalRequestsOutstanding</td>
<td>Number of requests currently proxied that have not yet returned</td>
</tr>
<tr>
<td>totalRequestsTimedOut</td>
<td>Number of requests that have timed out since last server restart or number requests not returned from proxy server within the [configured] initial timeout interval</td>
</tr>
<tr>
<td>totalRequestsAcknowledged</td>
<td>Number of responses received since last server restart</td>
</tr>
<tr>
<td>totalResponsesDroppedForNotInCache</td>
<td>Number of responses dropped because their ID did not match the ID of any Pending requests</td>
</tr>
<tr>
<td>totalResponsesDroppedForSignatureMismatch</td>
<td>Number of responses dropped because their response authenticator did not decode to the correct shared secret</td>
</tr>
<tr>
<td>totalRequestsDroppedAfterMaxTries</td>
<td>Number of requests dropped because no response was received after retrying the configured number of times. This value is different from totalRequestsTimedOut because using the default configuration values, no response within 2000 ms bumps the TimedOut counter, but it waits 14000 ms (2000 + 4000 + 8000) to bump this counter.</td>
</tr>
</tbody>
</table>
aregcmd Commands

Chapter 2    Using the aregcmd Commands

Stats for RADIUS Client
You can query the statistics of RADIUS clients using the following command:

`aregcmd stats /r/client/localhost`  

The following is an example of the statistical information provided for RADIUS clients after you issue the `stats` command:

```
Radius statistics for client: 127.0.0.1, localhost, ipv4
totalAuthAccessRequests = 15
totalAuthDupAccessRequests = 0
totalAuthAccessAccepts = 15
totalAuthAccessRejects = 0
totalAuthAccessChallenges = 0
totalAuthMalformedAccessRequests = 0
totalAuthBadAuthenticators = 0
totalAuthPacketsDropped = 0
totalAuthUnknownTypes = 0
totalAccPacketsDropped = 0
totalAccRequests = 0
totalAccDupRequests = 0
totalAccResponses = 0
totalAccBadAuthenticators = 0
totalAccMalformedRequests = 0
totalAccNoRecords = 0
totalAccUnknownTypes = 0
```

Table 2-2 lists the statistics displayed for the RADIUS client by the `stats` command and the meaning of the values.

<table>
<thead>
<tr>
<th>Stats Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>RADIUS Client statistics for:</td>
<td>Provides client’s IP address, name, and IP address type</td>
</tr>
<tr>
<td>totalAuthAccessRequests</td>
<td>Number of authentication access requests that are received by Prime Access Registrar from the client.</td>
</tr>
<tr>
<td>totalAuthDupAccessRequests</td>
<td>Number of duplicate authentication access requests that are received by Prime Access Registrar from the client.</td>
</tr>
<tr>
<td>totalAuthAccessAccepts</td>
<td>Number of authentication access requests from the client that are accepted by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAuthAccessRejects</td>
<td>Number of authentication access requests from the client that are rejected by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAuthAccessChallenges</td>
<td>Number of authentication challenges that are faced by Prime Access Registrar for the requests raised by the client.</td>
</tr>
</tbody>
</table>
status

Use the `aregcmd` command `status` to learn whether or not the specified server has been started. When the server is running, Prime Access Registrar displays its health.

The syntax is:

```
status
```

stop

Use the `aregcmd` command `stop` to cause the server to no longer accept requests.

The syntax is:
tacacs-stats

Use the aregcmd command tacacs-stats to provide statistical information of TACACS+.

The syntax is:

    tacacs-stats

The following is an example of the statistical information provided after you issue the tacacs-stats command:

Global Tacacs+ Statistics
    serverStartTime = Mon Apr 15 01:17:34 2013
    serverResetTime = Mon Apr 15 01:17:34 2013
    serverState = Running
    totalPacketsReceived = 60
    totalPacketsSent = 60
    totalRequests = 60
    totalResponses = 60
    totalAuthenticationRequests = 2
    totalAuthenticationAccepts = 2
    totalAuthenticationRejects = 0
    totalAuthenticationChallengeRequests = 0
    totalAuthenticationResponses = 2
    totalAuthorizationRequests = 56
    totalAuthorizationAccepts = 38
    totalAuthorizationRejects = 18
    totalAuthorizationResponses = 56
    totalAccountingRequests = 2
    totalAccountingAccepts = 2
    totalAccountingRejects = 0
    totalAccountingResponses = 2
    totalPacketsInUse = 0
    totalPacketsDropped = 0

See TACACSStatistics for more information.

tacacs-reset-stats

Use the aregcmd command tacacs-reset-stats to reset TACACS+ statistics displayed with the stats command. The tacacs-reset-stats command also resets SNMP counters.

The tacacs-reset-stats command provides a way of resetting the TACACS+ statistics without having to reload or restart the server.

The syntax is:

    tacacs-reset-stats

dia-stats

Use the aregcmd command dia-stats to provide statistical information of Diameter.

The syntax is:

    dia-stats
The following is an example of the statistical information provided for a Diameter remote server after you issue the following command:

```
dia-stats /Radius/RemoteServers/dia
```

Diameter Remote server statistics for: dia, 10.81.79.76, port 3868
active = TRUE
cDiaRemSvrRTTAverage = 25ms
cDiaRemSvrRTTDeviation = 0ms
cDiaRemSvrServerType = Diameter
cDiaRemSvrTotalRequestsPending = 0
cDiaRemSvrTotalRequestsOutstanding = 0
cDiaRemSvrTotalRequestsAcknowledged = 0
cDiaRemSvrStatsState = Closed
cDiaRemSvrStatsASRsIn = 0
cDiaRemSvrStatsASRsOut = 0
cDiaRemSvrStatsASAsIn = 0
cDiaRemSvrStatsASAsOut = 0
cDiaRemSvrStatsACRsIn = 0
cDiaRemSvrStatsACRsOut = 0
cDiaRemSvrStatsACAsIn = 0
cDiaRemSvrStatsACAsOut = 0
cDiaRemSvrStatsCERsIn = 0
cDiaRemSvrStatsCERsOut = 0
cDiaRemSvrStatsCEAsIn = 0
cDiaRemSvrStatsCEAsOut = 0
cDiaRemSvrStatsDWRsIn = 0
cDiaRemSvrStatsDWRsOut = 0
cDiaRemSvrStatsDWAsIn = 0
cDiaRemSvrStatsDWAsOut = 0
cDiaRemSvrStatsDPRsIn = 0
cDiaRemSvrStatsDPRsOut = 0
cDiaRemSvrStatsDPAsIn = 0
cDiaRemSvrStatsDPAsOut = 0
cDiaRemSvrStatsRARsIn = 0
cDiaRemSvrStatsRARsOut = 0
cDiaRemSvrStatsRAAsIn = 0
cDiaRemSvrStatsRAAsOut = 0
cDiaRemSvrStatsSTRsIn = 0
cDiaRemSvrStatsSTRsOut = 0
cDiaRemSvrStatsSTAsIn = 0
cDiaRemSvrStatsSTAsOut = 0
cDiaRemSvrStatsRedirectEvents = 0
cDiaRemSvrStatsAccDupRequests = 0
cDiaRemSvrStatsAccsNotRecorded = 0
cDiaRemSvrStatsWhoInitDisconnect = 0
cDiaRemSvrStatsAccRetrans = 0
cDiaRemSvrStatsTotalRetrans = 0
cDiaRemSvrStatsAccPendRequestsOut = 0
cDiaRemSvrStatsAccReqsDropped = 0
cDiaRemSvrStatsHBByHDropMessages = 0
cDiaRemSvrStatsEToEDupMessages = 0
cDiaRemSvrStatsUnknownTypes = 0
cDiaRemSvrStatsProtocolErrors = 0
cDiaRemSvrStatsTransientFailures = 0
cDiaRemSvrStatsPermanentFailures = 0
cDiaRemSvrStatsDWCurrentStatus = 0
cDiaRemSvrStatsTransportDown = 0
cDiaRemSvrStatsTimeoutConnAtmpts = 5
cDiaRemSvrStatsMARsIn = 0
cDiaRemSvrStatsMARsOut = 0
cDiaRemSvrStatsMAAsIn = 0
cDiaRemSvrStatsMAAsOut = 0
cDiaRemSvrStatsSARsIn = 0
cDiaRemSvrStatsSARsOut = 0
cDiaRemSvrStatsSAAAsIn = 0
cDiaRemSvrStatsSAAAsOut = 0
cDiaRemSvrStatsRTRsIn = 0
cDiaRemSvrStatsRTRsOut = 0
cDiaRemSvrStatsRTAsIn = 0
cDiaRemSvrStatsRTAsOut = 0
cDiaRemSvrStatsPPRsIn = 0
cDiaRemSvrStatsPPRsOut = 0
cDiaRemSvrStatsPPAsIn = 0
cDiaRemSvrStatsPPAsOut = 0
cDiaRemSvrStatsDERsIn = 0
cDiaRemSvrStatsDERsOut = 0
cDiaRemSvrStatsDEAsIn = 0
cDiaRemSvrStatsDEAsOut = 0
cDiaRemSvrStatsAARsIn = 0
cDiaRemSvrStatsAARsOut = 0
cDiaRemSvrStatsAAAsIn = 0
cDiaRemSvrStatsAAAsOut = 0

For detailed information about the Diameter remote server stats, see DiameterStatistics, page 3-138.

The following is an example of the statistical information provided for a Diameter peer after you issue the following command:

dia-stats /Radius/Clients/vm24

Diameter Peer statistics for: dia, 10.81.79.76, port 3868
cdbpPeerStatsState = Closed
cdbpPeerStatsASAsOut = 0
cdbpPeerStatsACRsIn = 0
cdbpPeerStatsACRsOut = 0
cdbpPeerStatsACAsIn = 0
cdbpPeerStatsACAsOut = 0
cdbpPeerStatsCERsIn = 1
cdbpPeerStatsCERsOut = 0
cdbpPeerStatsCEAsIn = 0
cdbpPeerStatsCEAsOut = 0
cdbpPeerStatsDWRsIn = 0
cdbpPeerStatsDWRsOut = 0
cdbpPeerStatsDWAsIn = 0
cdbpPeerStatsDWAsOut = 0
cdbpPeerStatsDPRsIn = 0
cdbpPeerStatsDPRsOut = 0
cdbpPeerStatsDPAsIn = 0
cdbpPeerStatsDPAsOut = 0
cdbpPeerStatsRARsIn = 0
cdbpPeerStatsRARsOut = 0
cdbpPeerStatsRAAsIn = 0
cdbpPeerStatsRAAsOut = 0
cdbpPeerStatsSTRsIn = 0
cdbpPeerStatsSTRsOut = 0
cdbpPeerStatsSTAsIn = 0
cdbpPeerStatsSTAsOut = 0
cdbpPeerStatsRedirectEvents = 0
cdbpPeerStatsAccDupRequests = 0
cdbpPeerStatsMalformedRequests = 0
cdbpPeerStatsAccsNotRecorded = 0
cdbPeerStatsWhoInitDisconnect = 0
cdbpPeerStatsAccRetrans = 0
cdbpPeerStatsTotalRetrans = 0
cdbpPeerStatsAccPendRequestsOut = 0
cdbpPeerStatsAccRequestsDropped = 0
cdbpPeerStatsHBByHDropMessages = 0
cdbpPeerStatsEToEDupMessages = 0
cdbpPeerStatsUnknownTypes = 0
cdbpPeerStatsProtocolErrors = 0
cdbpPeerStatsTransientFailures = 0
cdbpPeerStatsPermanentFailures = 0
cdbpPeerStatsDWCurrentStatus = 2
cdbpPeerStatsTransportDown = 0
cdbpPeerStatsTimeoutConnAtmpts = 0
cdbpPeerStatsASRsIn = 0
cdbpPeerStatsASRsOut = 0
cdbpPeerStatsASAsIn = 0
cdbpPeerStatsASAsOut = 0
cdbpPeerStatsMARsIn = 0
cdbpPeerStatsMARsOut = 0
cdbpPeerStatsMAAsIn = 0
cdbpPeerStatsMAAsOut = 0
cdbpPeerStatsSARsIn = 0
cdbpPeerStatsSARsOut = 0
cdbpPeerStatsSAAsIn = 0
cdbpPeerStatsSAAsOut = 0

dia-stats-reset

Use the aregcmd command **dia-stats-reset** to reset Diameter statistics displayed with the stats command. The **dia-stats-reset** command also resets SNMP counters.

The **dia-stats-reset** command provides a way of resetting the Diameter statistics without having to reload or restart the server.

The syntax is:

```
dia-stats-reset
```

trace

Use the aregcmd command **trace** to set the trace level in the specified server to a new value. The trace level governs how much information is displayed about the contents of a packet. When the trace level is zero, no tracing is performed. The higher the trace level, the more information displayed. The highest trace level currently used by the Prime Access Registrar server is trace level 5.
Although the highest `trace` level supported by the Prime Access Registrar server is `trace` level 5, an extension point script might use a higher level. There is no harm in setting the `trace` to a level higher than 5. However, increasing the trace level impacts the system performance.

The `trace` levels are inclusive, meaning that if you set `trace` to level 3, you will also get the information reported for `trace` levels 1 and 2. If you set trace level 4, you also get information reported for `trace` levels 1, 2, and 3.

When you do not specify a server, Prime Access Registrar sets the `trace` level for all of the servers in the current cluster. When you do not specify a value for the `trace` level, Prime Access Registrar displays the current value of the `trace` level. The default is 0.

The syntax for setting the `trace` level is:

```
trace [<server>] [<level>]
```

Table 2-4 lists the different `trace` levels and the information returned.

<table>
<thead>
<tr>
<th>Trace Level</th>
<th>Information Returned by Trace Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No trace performed.</td>
</tr>
<tr>
<td>1</td>
<td>Reports when a packet is sent or received or when there is a change in a remote server’s status.</td>
</tr>
<tr>
<td>2</td>
<td>Indicates the following:</td>
</tr>
<tr>
<td></td>
<td>- Which services and session managers are used to process a packet</td>
</tr>
<tr>
<td></td>
<td>- Which client and vendor objects are used to process a packet</td>
</tr>
<tr>
<td></td>
<td>- Detailed remote server information for LDAP and RADIUS, such as sending a packet and timing out</td>
</tr>
<tr>
<td></td>
<td>- Details about poorly formed packets</td>
</tr>
<tr>
<td></td>
<td>- Details included in trace level 1</td>
</tr>
<tr>
<td>3</td>
<td>Indicates the following:</td>
</tr>
<tr>
<td></td>
<td>- Error traces in TCL scripts when referencing invalid RADIUS attributes.</td>
</tr>
<tr>
<td></td>
<td>- Which scripts have been executed</td>
</tr>
<tr>
<td></td>
<td>- Details about local UserList processing</td>
</tr>
<tr>
<td></td>
<td>- Details included in trace levels 1 and 2</td>
</tr>
</tbody>
</table>
**trace-file-count**

Use the `aregcmd` command `trace-file-count` to change the trace log file count dynamically without requiring a server reload. The syntax is:

```
trace-file-count n
```

Where `n` is a number that specifies the number of trace log files. This function is helpful for debugging situations when you do not want to perform a `reload`.

**unset**

Use the `aregcmd` command `unset` to remove items from an ordered list. Specify the numeric index of the element to remove. When the ordered list is not the current context, specify the path to the list before specifying the numeric index.

When you remove an item from the list, Prime Access Registrar renumbers the list.

The syntax is:

```
unset [<path>/]<index>
```

This command applies to lists of servers by index, the Profiles attribute list, the Rules Attributes list, the enumeration list in the Attribute dictionary, and the LDAPtoRadiusMappings and LDAPtoEnvironmentMappings mappings.

**validate**

Use the `aregcmd` command `validate` to check the consistency and validity of the specified server’s configuration. If Prime Access Registrar discovers any errors, it displays them.

The syntax is:
OpenSSL Commands

This section contains a list of OpenSSL commands. You can use them on the command line or insert them into scripts.

This section contains the following topics:

- `ecparam`
- `req`
- `ca`

ecparam

Use the OpenSSL command `ecparam` to manipulate or generate elliptical curve (EC) parameter files. The syntax is:

```
ecparam
```
Chapter 2      Using the aregcmd Commands

aregcmd Command Logging

aregcmd now records the commands that are either entered interactively, on the command line, or executed in batch mode. The recorded commands are saved in the aregcmd_log file, which resides in the logs directory within the Prime Access Registrar installation directory.

For security reasons, aregcmd blocks out the actual password that is entered as part of the command and replaces it with <passwd>.

In interactive mode, aregcmd logs the actions that are taking place in the exit/logout dialog box. The action can be save or not save if the configuration database has been modified after the last execution of the save command.

In non-interactive (batch or command-line) mode, aregcmd replaces the empty field with a NULL string.

aregcmd is now installed as a setgid program where the group is set to staff. This allows a non-root user to run aregcmd while still being able to write to the aregcmd_log log file. During the installation of the Prime Access Registrar software, you are prompted whether you want to install aregcmd with setuid/setgid permissions. You must reply “yes” unless you only run aregcmd as user root.

The following is the format of an entry in the exit/logout dialog box when not save has been specified:

```
mm/dd/yyyy HH:MM:SS aregcmd Info Configuration 0 [username] ( **New config is not saved!...proceed to logout.)
```

The following is sample output of an entry in the exit/logout dialog box when not save has been specified:

```
10/12/2013 16:19:02 aregcmd Info Configuration 0 [localhost admin] --> *** New config is not saved!...proceed to logout.
```

The following is the format of an entry in the exit/logout dialog box when save has been specified:

```
mm/dd/yyyy HH:MM:SS aregcmd Info Configuration 0 [username] ( exit )
mm/dd/yyyy HH:MM:SS aregcmd Info Configuration 0 [username] ( *** New config saved!...proceed to logout.)
```

req

Use the OpenSSL command req to create and process certificate requests.

The syntax is:

```
req
```

can

Use the OpenSSL command ca used to sign certificate requests and generate CRLs it also maintains a text database of issued certificates and their status.

The syntax is:

```
ca
```
aregcmd Command Line Editing

Commands entered at the `aregcmd` prompt can be edited with a subset of the standard EMACS-style keystrokes. In addition, the command history can be accessed using the arrow keys on the keyboard. Use the Up arrow to retrieve the previous command and the Down arrow to retrieve the next command. A description of the supported key strokes are shown in Table 2-5.

<table>
<thead>
<tr>
<th>Key Stroke</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ctrl A</td>
<td>Go to the beginning of the line.</td>
</tr>
<tr>
<td>Ctrl B</td>
<td>Move back one character.</td>
</tr>
<tr>
<td>Ctrl D</td>
<td>Delete the character at the cursor.</td>
</tr>
<tr>
<td>Ctrl E</td>
<td>Go to the end of the line.</td>
</tr>
<tr>
<td>Ctrl F</td>
<td>Move forward one character.</td>
</tr>
<tr>
<td>Ctrl N</td>
<td>Retrieve the next line.</td>
</tr>
<tr>
<td>Ctrl P</td>
<td>Retrieve the previous line.</td>
</tr>
</tbody>
</table>

Table 2-5 aregcmd Command Line Editing Keystrokes

aregcmd Error Codes

Table 2-6 lists the error codes used in `aregcmd`.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>OK</td>
</tr>
<tr>
<td>300</td>
<td>Command failed to parse; usually an unbalanced quotation</td>
</tr>
<tr>
<td>301</td>
<td>Unknown command; usually caused by a misspelled command</td>
</tr>
<tr>
<td>302</td>
<td>Ambiguous command; characters you have entered select more than one command</td>
</tr>
<tr>
<td>303</td>
<td>Not logged in; you have logged out of <code>aregcmd</code> and attempted another <code>aregcmd</code> command</td>
</tr>
<tr>
<td>304</td>
<td>Too few arguments</td>
</tr>
<tr>
<td>305</td>
<td>Too many arguments</td>
</tr>
<tr>
<td>306</td>
<td>Invalid argument</td>
</tr>
<tr>
<td>307</td>
<td>Object not found or path ambiguous; you have tried to set an unknown object or provided a partial name that matches multiple objects</td>
</tr>
<tr>
<td>308</td>
<td>Object already exists</td>
</tr>
<tr>
<td>309</td>
<td>Confirmation password did not match; when setting a user password, the re-entered password did not match the initial entry</td>
</tr>
<tr>
<td>310</td>
<td>Command failed; a generic response for a command that failed for a reason other than those listed here</td>
</tr>
<tr>
<td>311</td>
<td>Command is interactive; possibly due to attempting to batch mode with an interactive command</td>
</tr>
<tr>
<td>312</td>
<td>Validation failed; a new or modified object is not valid</td>
</tr>
</tbody>
</table>

Table 2-6 aregcmd Error Codes
### Table 2-6  `aregcmd` Error Codes (continued)

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>313</td>
<td>Failed to reread; an error occurred while synchronizing changes from another <code>aregcmd</code> session; occurs only when using multiple <code>aregcmd</code> instances</td>
</tr>
<tr>
<td>314</td>
<td>Failed to open the pager; the PAGER environment variable is set to something that does not exist and the <code>stats</code> command is entered</td>
</tr>
<tr>
<td>315</td>
<td>Property is not modifiable; an administrator has attempted to modify a read-only property</td>
</tr>
<tr>
<td>316</td>
<td>Command failed: session is ViewOnly; an view-only administrator has attempted to modify a property</td>
</tr>
<tr>
<td>317</td>
<td>Command failed: server is a Replication Slave; an administrator has attempted to modify a replicated property on a slave of an SMDBR network</td>
</tr>
<tr>
<td>400</td>
<td>Login failed; an incorrect username or password was given during <code>aregcmd</code> log in</td>
</tr>
<tr>
<td>401</td>
<td>Unable to access server; <code>aregcmd</code> was unable to communicate with the <code>radius</code> process usually because the process is not running or is otherwise unresponsive</td>
</tr>
<tr>
<td>402</td>
<td>Login failed: version of aregcmd is incompatible with server; a new version of <code>aregcmd</code> has tried to connect with an older version of Prime Access Registrar</td>
</tr>
<tr>
<td>500</td>
<td>Internal error; a generic <code>aregcmd</code> error</td>
</tr>
</tbody>
</table>
CHAPTER 3

Using the Graphical User Interface

Cisco Prime Access Registrar (Prime Access Registrar) is a Remote Authentication Dial-In User Service (RADIUS) / Diameter server that enables multiple dial-in Network Access Server (NAS) devices to share a common authentication, authorization, and accounting database.

This chapter describes how to use the standalone graphical user interface (GUI) of Prime Access Registrar to:

- Configure Cisco Prime Access Registrar
- Manage Network Resources managed by Prime Access Registrar
- Administer Prime Access Registrar related activities

The following topics help you to work with and understand the Prime Access Registrar GUI:

- Launching the GUI
- Common Methodologies
- Dashboard
- Configuring Cisco Prime Access Registrar
- Network Resources
- Administration
- Read-Only GUI

Launching the GUI

Prime Access Registrar requires you to use Microsoft Internet Explorer 8.0 SP1 (Windows 2000 and Windows XP). You start the GUI by pointing your browser to the Prime Access Registrar server and port 8080, as in the following:

http://ar_server_name:8080

Note

You can also use Mozilla Firefox 16.0 and Google Chrome 23.0 browsers to launch the Prime Access Registrar GUI. It can be launched using IPv6 address also.

To start a secure socket layer (SSL) connection, use https to connect to the Prime Access Registrar server and port 8443, as in the following:

https://ar_server_name:8443
By default, both HTTP and HTTPS are enabled. The following sections describe how to disable HTTP and HTTPS:

- Disabling HTTP
- Disabling HTTPS

For proper function of Prime Access Registrar GUI, the DNS name resolution for the server’s hostname should be defined precisely.

### Disabling HTTP

To disable HTTP access, you must edit the `server.xml` file in the `/cisco-ar/apache-tomcat-7.0.42/conf` directory. You must have root privileges to edit this file.

Use a text editor such as `vi` to open the `server.xml` file, and comment out lines 96-99. Use the `<!--` character sequence to begin a comment. Use the `-->` character sequence to end a comment.

The following are lines 93-99 of the `server.xml` file:

```xml
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
<!-- CHANGE MADE: Note: to disable HTTP, comment out this Connector -->
<Connector port="8080" maxHttpHeaderSize="8192"
 maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
 enableLookups="false" redirectPort="8443" acceptCount="100"
 connectionTimeout="20000" disableUploadTimeout="true" />
```

The following example shows these lines with beginning and ending comment sequences to disable HTTP:

```xml
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
<!-- CHANGE MADE: Note: to disable HTTP, comment out this Connector -->
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080
<Connector className="org.apache.catalina.connector.http.HttpConnector"
 port="8080" minProcessors="5" maxProcessors="75"
 enableLookups="true" redirectPort="8443"
 acceptCount="10" debug="0" connectionTimeout="60000"/>
-->
```

After you modify the `server.xml` file, you must restart the Prime Access Registrar server for the changes to take effect. Use the following command line to restart the server:

```
/opt/CSCOar/bin/arserver restart
```

### Disabling HTTPS

To disable HTTPS access, you must edit the `server.xml` file in the `/cisco-ar/apache-tomcat-7.0.42/conf` directory. You must have root privileges to edit this file.

Use a text editor such as `vi` to open the `server.xml` file, and comment out lines 116-121. Use the `<!--` character sequence to begin a comment. Use the `-->` character sequence to end a comment.

The following are lines 111-121 of the `server.xml` file:

```xml
<!-- Define an SSL HTTP/1.1 Connector on port 8443 -->
<!-- CHANGE MADE: enabled HTTPS. -->
```

```
<!-- Define an SSL HTTP/1.1 Connector on port 8443
<Connector className="org.apache.tomcat.bio.SocketServerConnector"
 port="8443" maxConnections="100000" maxThreads="50" minSpareThreads="25"
 enableLookups="false" redirectPort="8443" acceptCount="100"
 connectionTimeout="20000" disableUploadTimeout="true" />
-->
```

After you modify the `server.xml` file, you must restart the Prime Access Registrar server for the changes to take effect. Use the following command line to restart the server:

```
```
```
Launching the GUI

Note: to disable HTTPS, comment out this Connector -->
<Connector port="8443" maxHttpHeaderSize="8192"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="true" disableUploadTimeout="true"
    acceptCount="100" scheme="https" secure="true"
    clientAuth="false"
    keystoreFile="/cisco-ar/certs/tomcat/server-cert.p12"
    keystorePass="cisco" keystoreType="PKCS12" sslProtocol="TLS" />
</Connector>

The following example shows these lines with beginning and ending comment sequences to disable HTTPS.

<!-- Define an SSL HTTP/1.1 Connector on port 8443 -->
<!-- CHANGE MADE: enabled HTTPS. Note: to disable HTTPS, comment out this Connector -->
<!--
<Connector className="org.apache.catalina.connector.http.HttpConnector"
    port="8443" minProcessors="5" maxProcessors="75"
    enableLookups="true"
    acceptCount="10" debug="0" scheme="https" secure="true">
    <Factory className="org.apache.catalina.net.SSLServerSocketFactory"
        keystoreFile="/cisco-ar/certs/tomcat/server-cert.p12"
        keystorePass="cisco" keystoreType="PKCS12"
        clientAuth="false" protocol="TLS"/>
</Connector>
-->

After you modify the `server.xml` file, you must restart the Prime Access Registrar server for the changes to take effect. Use the following command line to restart the server:

```
/opt/CSCOar/bin/arserver restart
```

Login Page

The login page has fields for a username and password. This page displays when you first attempt to log into the system, if a session times out, or after you log out of the system.

Logging In

Users who are configured as Administrators can log into the Prime Access Registrar server.

Note

While logging in, do not enable the save password option in the browser.

Logging in

To log into the Prime Access Registrar GUI:

**Step 1** Enter the relevant url in the browser. The Prime Access Registrar Login page is displayed.

**Step 2** Enter the credentials in the provided fields.
Step 3  
Click **Login**. The Prime Access Registrar main page is displayed.

---

**Note**  
After installation of Prime Access Registrar server, when you log into the application for the first time, the application redirects to the change password page.

---

**Refreshing the pages using the GUI**  
To stop the server (when it is running), and then immediately start the server, click the **Reload** link.

**Restarting the GUI**  
To restart the Prime Access Registrar server, click the **Restart** link.

---

**Note**  
If aregcmd interface is active, then it needs to be closed for restarting the Prime Access Registrar server.

---

Logging Out  
To log out of the Prime Access Registrar GUI, click **Logout** in the upper right portion of the Prime Access Registrar GUI window.

---

**Common Methodologies**  
This section explains the operations that are common across the GUI interface of Prime Access Registrar. The functions explained in this section are referred throughout to this help system.

This section describes the following:

- Filtering Records
- Deleting Records
- Setting Record Limits per Page
- Performing Common Navigations
- Relocating Records

---

**Filtering Records**  
To filter a record:

**Step 1**  
Navigate to the required page. For example, choose **Configuration > Profiles**. The Profile page is displayed.

**Step 2**  
Enter the known details of the record in the **Filter** text box.
Step 3 Click **Go**. The matching records are displayed in the search criteria below.

Step 4 Click **Clear Filter** to clear the performed filter.

You can also perform the following:

- Deleting Records
- Editing Records
- Setting Record Limits per Page
- Performing Common Navigations
- Relocating Records

## Editing Records

To edit the required records:

Step 1 Navigate to the required page.

Step 2 Search for a record using the filter option, if required.

Step 3 Choose the required record that you want to edit.

Step 4 Click **Edit**. The selected record details are displayed in the appropriate page.

Step 5 Make the necessary changes.

Step 6 Click **Submit** or **Update** to save the details. The page is displayed with the updated details and a message is prompted. Otherwise click **Cancel** to return to the page without saving the details.

You can also perform the following:

- Filtering Records
- Deleting Records
- Setting Record Limits per Page
- Performing Common Navigations
- Relocating Records

## Deleting Records

To delete a record:

Step 1 Navigate to the required page. For example, choose **Configuration > Profiles**. The Profile page is displayed.

Step 2 Search for a record using the filter option, if required.
Step 3 Check the check box against the record that you want to delete.

Step 4 Click Delete. A message is displayed on successful deletion of the record.

You can also perform the following:
- Filtering Records
- Editing Records
- Setting Record Limits per Page
- Performing Common Navigations
- Relocating Records

### Setting Record Limits per Page

To set the numbers of records to be displayed per page, select the record limit from the list available and click the Go button. The available denominations are 10, 25, 50, 100, and All.

You can also perform the following:
- Filtering Records
- Editing Records
- Deleting Records
- Performing Common Navigations
- Relocating Records

### Performing Common Navigations

On existence of more records that cannot be accommodated in a page, the records are displayed in multiple pages. Table 3-1 describes the icons used for page navigation.

**Table 3-1 Page Navigation Icons**

<table>
<thead>
<tr>
<th>Icons</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>☛</td>
<td>To view the next page</td>
</tr>
<tr>
<td>◀</td>
<td>To return back to previous page</td>
</tr>
<tr>
<td>◊</td>
<td>To view the last page</td>
</tr>
<tr>
<td>◌</td>
<td>To return to the first page</td>
</tr>
</tbody>
</table>
You can also perform the following:

- Filtering Records
- Editing Records
- Deleting Records
- Setting Record Limits per Page
- Relocating Records

## Relocating Records

Table 3-2 describes the icons used for relocating records.

<table>
<thead>
<tr>
<th>Icons</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="icon" /></td>
<td>To move a record from the Available List to the Selected List</td>
</tr>
<tr>
<td><img src="image2" alt="icon" /></td>
<td>To move a record from the Selected List to the Available List</td>
</tr>
<tr>
<td><img src="image3" alt="icon" /></td>
<td>To move all the records from the Available List to the Selected List</td>
</tr>
<tr>
<td><img src="image4" alt="icon" /></td>
<td>To move all the records from the Selected List to the Available List</td>
</tr>
<tr>
<td><img src="image5" alt="icon" /></td>
<td>To move the selected record one step above</td>
</tr>
<tr>
<td><img src="image6" alt="icon" /></td>
<td>To move the selected record one step below</td>
</tr>
<tr>
<td><img src="image7" alt="icon" /></td>
<td>To move the selected record to the first position</td>
</tr>
<tr>
<td><img src="image8" alt="icon" /></td>
<td>To move the selected record to the last position</td>
</tr>
</tbody>
</table>

You can also perform the following:

- Filtering Records
- Editing Records
- Deleting Records
- Setting Record Limits per Page
- Performing Common Navigations
Dashboard

The dashboard of the Prime Access Registrar GUI shows you the overview on the status on the server and user session details. It consists of the three tabs: Server Status, User Sessions, and System Information.

The Server Status provides the following details:

- AAA Server status— includes the AAA Process, Process ID, and Status.
- Health status of the AAA Server— the status of the AAA Server with respect to the performance condition is displayed.

The User Sessions consists of two graphs.

- Number of Sessions versus Duration in Days
- Number of Sessions versus Duration in Weeks

The Number of Sessions vs Duration in Weeks report provides the session details with respect to the number of weeks for which it is queried. The Number of Sessions vs Duration in Days report provides the session details with respect to the number of days for which it is queried. The Time(mins) vs Username report provides the accumulated time with respect to the selected username. This report can also be viewed in the form of chart and grid. Click the relevant icons below the graph to view the details in the respective formats.

The System Information section consists of two graphs:

- Disk Availability for Prime Access Registrar Directory
- CPU Utilization

The Disk Availability for Prime Access Registrar Directory report provides the details of the available disk space and used disk space in the Prime Access Registrar directory. When you hover the mouse on the pie chart, the details of the disk space are displayed. The CPU Utilization report provides the utilization of the CPU for a specific time. The CPU usage is represented in kilobits per seconds.

Sessions

The Sessions feature of the dashboard helps you in viewing the records based on session id. Table 3-3 lists and describes the various session views in the page.

<table>
<thead>
<tr>
<th>Table 3-3</th>
<th>Different Session Views</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fields</td>
<td>Description</td>
</tr>
<tr>
<td>Release</td>
<td>Click to release the selected session details.</td>
</tr>
<tr>
<td>Release All</td>
<td>Click to release all the records from the list.</td>
</tr>
<tr>
<td>Send CoA</td>
<td>Click to send the CoA packet to the client device.</td>
</tr>
<tr>
<td>SendPoD</td>
<td>Click to send the disconnect packet to the NAS to clear sessions and an Accounting-Stop notification to the client listed in the session record.</td>
</tr>
<tr>
<td>Query All Sessions</td>
<td>Click to query all the sessions in the server.</td>
</tr>
</tbody>
</table>
To view sessions details:

**Step 1** Choose **Dashboard > Sessions**. The Sessions page appears.

**Step 2** Choose the required session id to view **Release, Release All, Send CoA, Send PoD,** and **Query All Session** details. The session details are displayed as described in the above table.

**Note** You can locate the session id using the filter option. See **Filtering Records** for more details.

---

### Configuring Cisco Prime Access Registrar

Prime Access Registrar’s operation and configuration are based on a set of objects. On configuring the Prime Access Registrar major components, the server objects can be created. These objects include the following:

- **RADIUS** — the root of the configuration hierarchy
- **Profiles** — contains individual Profiles
- **UserGroups** — contains individual UserGroups
- **UserList** — contains individual UserLists which in turn contain users
- **Users** — contains individual authentication or authorization details of a user
- **Scripts** — contains individual Scripts
- **Policies** — contains a set of rules applied to an Access-Request
- **Services** — contains individual Services
- **CommandSets** — contains commands and the action to perform during Terminal Access Controller Access-Control System Plus (TACACS+) command authorization
- **DeviceAccessRules** — contains conditions or expressions and the applicable command sets for TACACS+ command authorization
- **FastRules** — provides a mechanism to easily choose the right authentication, authorization, accounting, and query service(s), drop, reject, or break flows, choose session manager or other rules required for processing a packet
- **Replication** — maintains identical configurations on multiple machines simultaneously
- **RADIUSDictionary** — passes information between a script and the RADIUS server, or between scripts running on a single packet
- **VendorDictionary** — allows to maintain the attributes of the vendor with respect to vendor id, vendor type and the attributes required to support the major NAS
- **Vendor Attributes** — communicates prepaid user balance information from the Prime Access Registrar server to the AAA client, and actual usage, either interim or total, between the NAS and the Prime Access Registrar server
- **Vendors** — contains individual Vendors
- **Translations** — adds new attributes to a packet or change an existing attribute from one value to another.
- **TranslationGroups** — add translation groups for different user groups
- **SessionManagers** — contains individual Session Managers
- **ResourceManager** — contains individual Resource Managers
- **Remote Servers** — contains individual Remote Servers
- **Diameter** — contains Session Management, Applications, Commands, Diameter Attributes
- **Rules** — allows to set rules for service selection

**RADIUS**

The **Radius** object is the root of the hierarchy. For each installation of the Cisco Prime Access Registrar server, there is one instance of the **Radius** object. You reach all other objects in the hierarchy from the **Radius**.

Table 3-4 lists and describes the fields in the Radius Properties page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the list of servers in the cluster.</td>
</tr>
<tr>
<td>Version</td>
<td>Required; the currently installed version of Prime Access Registrar.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the server.</td>
</tr>
<tr>
<td>DefaultSessionManager</td>
<td>Cisco Prime Access Registrar uses this property if none of the incoming scripts sets the environment dictionary variable <strong>Session-Manager</strong>. This field is mandatory if you are upgrading to a later version of Prime Access Registrar.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; if there is a script, it is the first script Cisco Prime Access Registrar runs when it receives a request from any client and/or for any service.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; if there is a script, it is the last script Cisco Prime Access Registrar runs before it sends a response to any client.</td>
</tr>
<tr>
<td>DefaultAuthenticationService</td>
<td>Optional; Cisco Prime Access Registrar uses this property when none of the incoming scripts sets the environment dictionary variable <strong>Authentication-Service</strong>.</td>
</tr>
<tr>
<td>DefaultAuthorizationService</td>
<td>Optional; Cisco Prime Access Registrar uses this property when none of the incoming scripts sets the environment dictionary variable <strong>Authorization-Service</strong>.</td>
</tr>
</tbody>
</table>
Setting Up or Changing the Radius Properties

To set or change the Radius properties:

Step 1  Choose Configuration > Radius. The Radius Properties page appears.
Step 2  Specify the relevant details.
Step 3  Click Save to save the changes made to the Radius properties page.

On successful setting up of the RADIUS, a message is displayed.

Profiles

You use Profiles to group RADIUS attributes that belong together, such as attributes that are appropriate for a particular class of PPP or Telnet user. You can reference profiles by name from either the UserGroup or the User properties. Thus, if the specifications of a particular profile change, you can make the change in a single place and have it propagated throughout your user community.

Although you can use UserGroups or Profiles in a similar manner, choosing whether to use one rather than the other depends on your site. When you require some choice in determining how to authorize or authenticate a user session, then creating specific profiles, and creating a group that uses a script to choose among them is more flexible.

In such a situation, you might create a default group, and then write a script that selects the appropriate profile based on the specific request. The benefit to this technique is each user can have a single entry, and use the appropriate profile depending on the way they log in.

Table 3-5 lists and describes the fields in the Add Profiles page.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Profiles list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the profile.</td>
</tr>
<tr>
<td>RADIUS</td>
<td>Optional; set Radius, if the attribute and value need to be defined for RADIUS.</td>
</tr>
</tbody>
</table>
Configuring Cisco Prime Access Registrar

You can use the Profiles page for the following:

- Filtering Records
- Adding Profile Details
- Editing Records
- Deleting Records

Adding Profile Details

To add new profile details:

1. Choose **Configuration > Profiles**. The Profiles page is displayed.
2. Click **Add**. The Add Profile page is displayed.
3. Specify the required details.
4. Click **Submit** to save the specified details in the Profiles page. Otherwise click **Cancel** to return to the Profiles page without saving the details. On successful creation of the profiles, the Profiles page is displayed else a respective error message is displayed.

UserGroups

The **UserGroups** objects allow you to maintain common authentication and authorization attributes in one location, and then have many users reference them. By having a central location for attributes, you can make modifications in one place instead of having to make individual changes throughout your user community.

For example, you can use several **UserGroups** to separate users by the services they use, such as a group specifying PPP and another for Telnet.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VENDOR</td>
<td>Optional; set Vendor, if the attribute and value need to be defined for Vendor.</td>
</tr>
<tr>
<td>DIAMETER</td>
<td>Optional; set Diameter, if the attribute and value need to be defined for Diameter.</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Optional; based on the Attribute Type selected, the attribute name is automated. Set the relevant name for the attribute type selected.</td>
</tr>
<tr>
<td>Value Attribute</td>
<td>Optional; set the value for the selected attribute. Click the <strong>Add</strong> button to save the details and list it in Radius and Value list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See <strong>Relocating Records</strong> for more details. To delete the available attributes, select the relevant attribute and click the <strong>Delete</strong> button below.</td>
</tr>
</tbody>
</table>

You can use the Profiles page for the following:

- Filtering Records
- Adding Profile Details
- Editing Records
- Deleting Records
Table 3-6 lists and describes the fields in the Add User Groups page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General Properties tab</strong></td>
<td></td>
</tr>
<tr>
<td>UserGroup Name</td>
<td>Required; must be unique in the <strong>UserGroup</strong> list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the group.</td>
</tr>
<tr>
<td>BaseProfile</td>
<td>Optional; when you set this to the name of a profile, Cisco Prime Access Registrar adds the properties in the Profile to the response dictionary as part of the authorization.</td>
</tr>
<tr>
<td>AuthenticationScript</td>
<td>Optional; when you set this property to the name of a script, you can use the Script to perform additional authentication checks to determine whether to accept or reject the user.</td>
</tr>
<tr>
<td>AuthorizationScript</td>
<td>Optional; when you set this property to the name of a script, you can use the script to add, delete, or modify the attributes of the Response dictionary.</td>
</tr>
<tr>
<td><strong>Attribute List tab</strong></td>
<td></td>
</tr>
<tr>
<td>RADIUS</td>
<td>Optional; set Radius, if the attribute and value need to be defined for RADIUS.</td>
</tr>
<tr>
<td>VENDOR</td>
<td>Optional; set Vendor, if the attribute and value need to be defined for Vendor.</td>
</tr>
<tr>
<td>DIAMETER</td>
<td>Optional; set Diameter, if the attribute and value need to be defined for Diameter.</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Optional; based on the Attribute Type selected, the attribute name is automated. Set the relevant name for the attribute type selected.</td>
</tr>
<tr>
<td>Attribute Value</td>
<td>Optional; set the value for the selected attribute. Click the <strong>Add</strong> button to save the details and list it in Name and Value list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the <strong>Delete</strong> button below.</td>
</tr>
<tr>
<td><strong>CheckItems List tab</strong></td>
<td></td>
</tr>
<tr>
<td>RADIUS</td>
<td>Optional; set Radius, if the attribute and value need to be defined for RADIUS.</td>
</tr>
<tr>
<td>VENDOR</td>
<td>Optional; set Vendor, if the attribute and value need to be defined for Vendor.</td>
</tr>
<tr>
<td>DIAMETER</td>
<td>Optional; set Diameter, if the attribute and value need to be defined for Diameter.</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Optional; based on the Attribute Type selected, the attribute name is automated. Set the relevant name for the attribute type selected.</td>
</tr>
<tr>
<td>Attribute Value</td>
<td>Optional; set the value for the selected attribute. Click the <strong>Add</strong> button to save the details and list it in Check Name and Check Value list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the <strong>Delete</strong> button below.</td>
</tr>
</tbody>
</table>
You can use the User Groups page for the following:

- Filtering Records
- Adding UserGroup Details
- Editing Records
- Deleting Records

**Adding UserGroup Details**

To add new user groups details:

**Step 1** Choose **Configuration > UserGroups**. The User Groups page is displayed.

**Step 2** Click **Add** to add new user group details. The Add UserGroup page is displayed.

**Step 3** Specify the required details.

**Step 4** Click **Submit** to save the specified details in the User Groups page. Otherwise click **Cancel** to return to the User Groups page without saving the details.

On successful creation of the user groups, the User Groups page is displayed else a respective error message is displayed.

**UserList**

The UserLists object contains all of the individual UserLists, which in turn, contain the specific users stored within Prime Access Registrar. Prime Access Registrar references each specific UserList by name from a Service whose type is set to local. When Prime Access Registrar receives a request, it directs it to a Service. When the Service has its type property set to local, the Service looks up the user’s entry in the specific UserList and authenticates and/or authorizes the user against that entry.

You can have more than one UserList in the UserLists object. Therefore, use the UserLists object to divide your user community by organization. For example, you might have separate UserLists objects for Company A and B, or you might have separate UserLists objects for different departments within a company.

Using separate UserLists objects allows you to have the same name in different lists. For example, if your company has three people named Bob and they work in different departments, you could create a UserList for each department, and each Bob could use his own name. Using UserLists lets you avoid the problem of Bob1, Bob2, and so on.

If you have more than one UserList, Prime Access Registrar can run a script in response to requests. The script chooses the Service, and the Service specifies the actual UserList which contains the user. The alternative is dynamic properties.
Note

The attributes defined for a user list must match the protocol of the incoming packet. For example, if the incoming packet is a Diameter packet, the attributes defined must be specific to Diameter or common to both RADIUS and Diameter. Similarly, if the incoming packet is a RADIUS packet, the attributes defined must be specific to RADIUS or common to both RADIUS and Diameter. Otherwise, the incoming packet will not be processed.

Table 3-7 lists and describes the fields in the Add User List page.

Table 3-7 User List Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserList Name</td>
<td>Required; must be unique.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the user list.</td>
</tr>
</tbody>
</table>

You can use the User List page for the following:

- Filtering Records
- Adding UserList Details
- Editing Records
- Deleting Records

Adding UserList Details

To add new user list details:

Step 1 Choose Configuration > UserList. The User List page is displayed.
Step 2 Click Add to add new user list details. The Add UserList page is displayed.
Step 3 Enter the required details.
Step 4 Click Submit to save the specified details in the User List page. Otherwise click Cancel to return to the User List page without saving the details.

On successful creation of the user list, the User List page is displayed else a respective error message is displayed.

Note After adding a new user list, you can add users to the user list. See Adding User Details for more information.
Users

The user objects are created to hold the necessary details to authenticate or authorize a user. These users form the component of User Lists, where their details are stored within Prime Access Registrar. The users in local Userlist can have multiple profiles.

Note

Usernames might not include the forward slash (/) character. If the Prime Access Registrar server receives an access request packet with a Username attribute containing a forward slash character and the Prime Access Registrar server uses an internal UserList to look up users, the server produces an error (AX_EINVAL) and might fail. If usernames require a forward slash, use a script to translate the slash to an acceptable, unused character.

Table 3-8 lists and describes the fields in the Add Users page.

Table 3-8  Users Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General Properties tab</strong></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Required; must be unique.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Required; must be checked to allow user access. If Enabled is not checked, user is denied access.</td>
</tr>
<tr>
<td>Allow Null Pwd</td>
<td>During authentication, if the Allow NULL Password environment variable is set to TRUE, user authentication is bypassed. By default, the Allow NULL Password environment variable is not set.</td>
</tr>
<tr>
<td>UserGroup</td>
<td>Use the drop-down list to select a UserGroup and use the properties specified in the UserGroup to authenticate and/or authorize the user. The default is none.</td>
</tr>
<tr>
<td>Password</td>
<td>Required; length must be between 0-253 characters.</td>
</tr>
<tr>
<td>Base Profile</td>
<td>Optional; use the drop-down list to select a Profile. If the service-type is not equal to Authenticate Only, Prime Access Registrar adds the properties in the Profile to the Response dictionary as part of the authorization. This field is optional for the CLI, but required for the GUI. Use the menu to select a profile other than the default None.</td>
</tr>
<tr>
<td>Confirm Password</td>
<td>Required; must match password.</td>
</tr>
<tr>
<td>User Defined</td>
<td>Optional; you can use this property to store notational information which you can then use to filter the UserList. This property also sets the environment variable for UserDefined.</td>
</tr>
<tr>
<td>Authentication Script</td>
<td>Optional; use the drop-down list to select the name of a script to perform additional authentication checks to determine whether to accept or reject the user. This field is optional for the CLI, but required for the GUI. Use the menu to select an Authentication Script other than the default None.</td>
</tr>
<tr>
<td>Authorization Script</td>
<td>Optional; use the drop-down list to select the name of a script to add, delete, or modify the attributes of the Response dictionary. This field is optional for the CLI, but required for the GUI. Use the menu to select an Authorization Script other than the default None.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the user.</td>
</tr>
<tr>
<td><strong>Attribute List tab</strong></td>
<td></td>
</tr>
</tbody>
</table>
Chapter 3     Using the Graphical User Interface

Configuring Cisco Prime Access Registrar

You can use the Users page for the following:

• Filtering Records
• Adding User Details
• Editing Records
• Deleting Records

### Adding User Details

To add new user details:

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Choose Configuration &gt; UserList. The User List page is displayed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 2</td>
<td>Click the user list name link. The Users page is displayed.</td>
</tr>
<tr>
<td>Step 3</td>
<td>Click Add to add new user details. The Add Users page is displayed.</td>
</tr>
<tr>
<td>Step 4</td>
<td>Specify the required details.</td>
</tr>
<tr>
<td>Step 5</td>
<td>Click Submit to save the specified details in the Users page. Otherwise click Cancel to return to the Users page without saving the details.</td>
</tr>
</tbody>
</table>

On successful creation of the user details, the Users page is displayed else a respective error message is displayed.

### Table 3-8   Users Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RADIUS</td>
<td>Optional; set Radius, if the attribute and value need to be defined for RADIUS.</td>
</tr>
<tr>
<td>VENDOR</td>
<td>Optional; set Vendor, if the attribute and value need to be defined for Vendor.</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Optional; based on the Attribute Type selected, the attribute name is automated. Set the relevant name for the attribute type selected.</td>
</tr>
<tr>
<td>Attribute Value</td>
<td>Optional; set the value for the selected attribute. Click the Add button to save the details and list it in Name and Value list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CheckItems List tab</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RADIUS</td>
<td>Optional; set Radius, if the attribute and value need to be defined for RADIUS.</td>
</tr>
<tr>
<td>VENDOR</td>
<td>Optional; set Vendor, if the attribute and value need to be defined for Vendor.</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Optional; based on the Attribute Type selected, the attribute name is automated. Set the relevant name for the attribute type selected.</td>
</tr>
<tr>
<td>Attribute Value</td>
<td>Optional; set the value for the selected attribute. Click the Add button to save the details and list it in Check Name and Check Value list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
</tbody>
</table>
Scripts

The **Script** objects define the function Cisco Prime Access Registrar invokes whenever the **Script** is referenced by name from other objects in the configuration.

There are four types of scripts:

- REX (RADIUS EXtension) scripts are written in C or C++, and thus are compiled functions that reside in shared libraries.
- TCL scripts are written in TCL, and are interpreted functions defined in source files.
- Java scripts
- Internal scripts, which allow you to add, modify, or delete attributes in the request, response, and environment dictionaries for RADIUS, Diameter, and TACACS+. For more information about internal scripts, see **Internal Scripts, page 11-14**.

When you use a Prime Access Registrar file service, Prime Access Registrar automatically closes any opened files. However, if you write scripts that manipulate files, you are responsible for closing them.

If you have more than one extension point script (defined under `/Radius/Scripts`) using the same Java class, only one instance of the class is created and used for all the extension point scripts.

For more information about scripts, see Chapter 11, “Using Extension Points.”

Table 3-9 lists and describes the fields in the Add Scripts page.

**Table 3-9 Script Object Properties**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Script Name</td>
<td>Required; must be unique in the Scripts list.</td>
</tr>
<tr>
<td>Language</td>
<td>Required; specify either REX, TCL, Java, or Internal.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the script.</td>
</tr>
<tr>
<td>File/Class Name</td>
<td>Required; specifies either a relative or absolute path. When you specify a relative path, the path must be relative to the <code>$INSTALL/scripts/radius/$Language</code> directory. When you specify an absolute path, the server must be able to reach it. For Java language scripts, the name of the class that implements the extension interface; the <code>.class</code> file should be placed in <code>/cisco-ar/scripts/radius/java</code></td>
</tr>
<tr>
<td>Entry Point</td>
<td>Required; when not set, Prime Access Registrar uses the value specified in the Name property.</td>
</tr>
<tr>
<td>Init Entry Point</td>
<td>Optional; if set, it must be the name of the global symbol Prime Access Registrar should call when it initializes the shared library at system start up, and just before it unloads the shared library.</td>
</tr>
</tbody>
</table>
Chapter 3      Using the Graphical User Interface

Configuring Cisco Prime Access Registrar

Table 3-9  Script Object Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Init Entry Point Arg</td>
<td>Optional; when set, it provides the arguments to be passed to the InitEntryPoint in the environmental variable Arguments.</td>
</tr>
<tr>
<td>Note</td>
<td>The InitEntryPoint properties allow you to perform initialization before processing and then cleanup before stopping the server. For example, when Prime Access Registrar unloads the script (when it stops the RADIUS server) it calls the InitEntryPoint again to allow it to perform any clean-up operations as a result of its initialization. One use of the function might be to allow the script to close an open Accounting log file before stopping the RADIUS server.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Required; select RADIUS or Diameter to indicate the protocol for which the attributes are to be modified.</td>
</tr>
<tr>
<td>ActionStatements</td>
<td>Select one of following the options:</td>
</tr>
<tr>
<td></td>
<td>• Simple Attribute Operation—allows you to add, modify, or delete an attribute value to the request, response, or environment dictionary</td>
</tr>
<tr>
<td></td>
<td>• Copy an Attribute—allows you to copy an attribute value from one dictionary to another</td>
</tr>
<tr>
<td></td>
<td>• Concatenate Operation—allows you to concatenate an attribute value from one dictionary to another</td>
</tr>
<tr>
<td></td>
<td>• Replace Operation—allows you to replace an attribute value from one dictionary to another</td>
</tr>
<tr>
<td></td>
<td>• Value Based Manipulations—allows you to manipulate attribute values in a dictionary based on a given text</td>
</tr>
<tr>
<td></td>
<td>• Log or Trace Messages—allows you to create different levels of log or trace messages</td>
</tr>
<tr>
<td></td>
<td>• I can do it myself—allows you to create your own script for the selected protocol</td>
</tr>
<tr>
<td>Left Side of Statement</td>
<td></td>
</tr>
<tr>
<td>Operation</td>
<td>Choose the operation to perform as Add, Modify, or Delete.</td>
</tr>
<tr>
<td>Dictionary</td>
<td>Choose Request, Response, or Environment to specify the RADIUS dictionary to apply the action to.</td>
</tr>
<tr>
<td>Attr Type</td>
<td>Applicable for RADIUS protocol; select RADIUS or VENDOR to indicate the attribute type.</td>
</tr>
<tr>
<td>Group AVP</td>
<td>Applicable for Diameter protocol; select the group AVP and its level to apply the action to.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Based on the attribute type/group AVP selected, choose the appropriate attribute to apply the action to.</td>
</tr>
<tr>
<td>Env Attribute</td>
<td>Enter the environment attribute to apply the action to.</td>
</tr>
<tr>
<td>This field is available only if the Dictionary chosen is Environment.</td>
<td></td>
</tr>
<tr>
<td>Right Side of Statement</td>
<td></td>
</tr>
</tbody>
</table>
Table 3-9  Script Object Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
</table>
| Text   | Enter the text that needs to be added, modified, or deleted to/from the given attribute in the selected dictionary. Only this field is available if the action statement is **Simple Attribute Operation** or **Replacement Operation**. This field is also available under the following circumstances:  
  - If the action statement is **Copy an Attribute**, **Concatenate Operation**, or **Value Based Manipulations**, and if the type is chosen as **Custom Text** |
| Type   | Select **Radius**, **Diameter**, or **Custom Text**. |
| Dictionary | If the type is set as Radius or Diameter, choose **Request**, **Response**, or **Environment** to specify the dictionary to apply the action to. |
| Attr Type | Applicable for RADIUS protocol; select **RADIUS** or **VENDOR** to indicate the attribute type. |
| Group AVP | Applicable for Diameter protocol; select the group AVP and its level to apply the action to. |
| Attribute | Based on the attribute type/group AVP selected, choose the appropriate attribute to apply the action to. |
| Env Attribute | Enter the environment attribute to apply the action to. If the Dictionary chosen is **Environment**. |

**Concatenate / Replace Options**

This section is available if the Action Statements field is set to **Concatenate Operation** or **Replace Operation**.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Select <strong>Radius</strong>, <strong>Diameter</strong>, or <strong>Custom Text</strong>.</td>
</tr>
<tr>
<td>Text</td>
<td>Enter the text to concatenate to or replace the given attribute value in the selected dictionary. Only this field is available if the action statement is <strong>Replace Operation</strong>. This field is also available if the action statement is <strong>Concatenate Operation</strong> and if the type is chosen as <strong>Custom Text</strong></td>
</tr>
<tr>
<td>Dictionary</td>
<td>If the type is Radius, choose <strong>Request</strong>, <strong>Response</strong>, or <strong>Environment</strong> to specify the RADIUS dictionary to apply the action to.</td>
</tr>
<tr>
<td>Attr Type</td>
<td>Applicable for RADIUS protocol; select <strong>RADIUS</strong> or <strong>VENDOR</strong> to indicate the attribute type.</td>
</tr>
<tr>
<td>Group AVP</td>
<td>Applicable for Diameter protocol; select the group AVP and its level to apply the action to.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Based on the attribute type/group AVP selected, choose the appropriate attribute to apply the action to.</td>
</tr>
<tr>
<td>Env Attribute</td>
<td>Enter the environment attribute to apply the action to. If the Dictionary chosen is <strong>Environment</strong>.</td>
</tr>
</tbody>
</table>

**Text Manipulations**

This section is available if the Action Statements field is set to **Value Based Manipulations**.
You can use the Scripts page for the following:

- Filtering Records
- Adding Script Details
- Editing Records
- Deleting Records

### Adding Script Details

To add new script details:

1. Choose **Configuration** > **Scripts**. The Scripts page is displayed.
2. Click **Add** to add new scripts details. The Script Details page is displayed.
3. Enter the required details.
4. Click **Save** to save the specified details in the Scripts page. Otherwise click **Cancel** to return to the Scripts page without saving the details.

On successful creation of the scripts, the Scripts page is displayed else a respective error message is displayed.
Policies

A Policy is a set of rules applied to an Access-Request. Table 3-10 lists and describes the fields in the Add Policies page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Policies list</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the Policy</td>
</tr>
<tr>
<td>Rules/Policies</td>
<td>Required; set the rules/policies to be grouped.</td>
</tr>
<tr>
<td>Operators</td>
<td>Required; set the operators to be grouped along with selected rules/policies. The selected rules and operators will be grouped and listed in the Grouping Box. To delete the available groups, select the relevant group from the Grouping list and click the Delete button below.</td>
</tr>
</tbody>
</table>

You can use the Policies page for the following:

- Filtering Records
- Adding Policy Details
- Editing Records
- Deleting Records

Adding Policy Details

To add new policy details:

**Step 1** Choose Configuration > Policies. The Policies page is displayed.

**Step 2** Click Add to add new policy details. The Policy Details page is displayed.

**Step 3** Specify the required details.

**Step 4** Click Submit to save the specified details in the Policies page. Otherwise click Cancel to return to the Policies page without saving the details.

On successful creation of the policies, the Policies page is displayed else a respective error message is displayed.

Services

Cisco Prime Access Registrar supports authentication, authorization, and accounting (AAA) services. In addition to the variety of built-in AAA services (specified in the Type property), Cisco Prime Access Registrar also enables you to add new AAA services through custom shared libraries.
This section lists the types of services available in Prime Access Registrar with their required and optional properties. The service you specify determines what additional information you must provide. The various types of services are:

- **Simple Services**
- **ServiceWithRS**
- **PEAP Service**
- **EAP Service**
- **Diameter Service**

## Simple Services

Prime Access Registrar provides the following simple services:

- **Rex**
- **File**
- **Trusted-ID**
- **Group**
- **Local**
- **Java**
- **WiMAX**
- **RADIUS-Query**
- **Dyn-Authz**
- **Diameter-RADIUS**
- **RADIUS-Diameter**
- **Diameter-Query**
- **3GPPAuthorization**
- **3GPP-Reverse-Authorization**

### Rex

Select rex service when a custom service needs to be created and a script for authentication, authorization, or accounting has to be used.

### File

Select File type when local accounting is to be performed using a specific file. The files under the configuration will be saved in the configured name when the server is invoked even if the service is not being invoked by any request packets.

Prime Access Registrar flushes the accounting record to disk before it acknowledges the request packets. Based on the specified maximum file size and age, it closes the accounting file, moves it to a new name, and reopens the file as a new file. The file names are based on its creation and modification dates.
**Trusted-ID**

Select the trusted-id service type to authorize and authenticate a user based on a Trusted ID. Using SSG's Transparent Auto-Login (TAL) feature, a TAL access-request packet contains a Trusted ID, such as a MAC address, that identifies the user without the user's real username and password. If Prime Access Registrar knows the user associated with the Trusted ID, it uses the Trusted ID to authenticate and authorize the user. For more information, see Chapter 15, “Using Trusted ID Authorization with SESM.”

**Group**

A group service contains a list of references to other services and specifies whether the responses from each of the services should be handled as a logical AND or OR function, which is specified in the Result-Rule attribute of Group Services. The default value is AND.

When the Result-Rule attribute is set to AND or OR, each referenced service is accessed sequentially, and the Group Service waits for a response from the first referenced service before moving on to the next service (if necessary).

The ResultRule settings parallel-and and parallel-or are similar to the AND and OR settings except that they ask each referenced service to process the request simultaneously instead of asking each referenced server sequentially, thereby saving processing time.

**Local**

Select local services when authentication and authorization needs to be performed by Prime Access Registrar server using a specific UserList.

**Java**

Select Java service type when a custom service needs to be created and to use an extension point script to provide the service’s functionality and handle both RADIUS and TACACS requests for authentication, authorization, or accounting.

**WiMAX**

Prime Access Registrar uses the Extensible Authentication Protocol (EAP) to enable the WiMAX feature. It captures the IP attributes and Mobility Keys that are generated during network access authentication.

**RADIUS-Query**

Select this service type to query cached data through RADIUS Packets. It contains the list of session managers to be queried from and a list of (cached) attributes to be returned in the Access-Accept packet in response to a RADIUS Query request. It is initiated through an extension point script or through the Rule and Policy Engine by setting it to a new environment variable named Query-Service.

**Dyn-Authz**

Select this service type to process dynamic authorization requests. This involves Change of Authorization (COA) and Packet of Disconnect (POD) features. For more information about these features, see Chapter 17, “Using Cisco Prime Access Registrar Server Features.”
Diameter-RADIUS

Select this service for Diameter to RADIUS translation to translate incoming Diameter request to a RADIUS equivalent and then the RADIUS response to Diameter equivalent. Prime Access Registrar provides scripting points, which operate on the original packet and on the newly translated packet based on request and response mapping. For more information, see Chapter 8, “Diameter.”

RADIUS-Diameter

Select this service for RADIUS to Diameter translation to translate incoming RADIUS request to a Diameter equivalent and then the Diameter response to RADIUS equivalent. Prime Access Registrar provides scripting points, which operate on the original packet and on the newly translated packet based on request and response mapping. For more information, see Chapter 8, “Diameter.”

Diameter-Query

Select this service type to query cached data through Diameter Packets. It contains the list of session managers to be queried from and a list of (cached) attributes to be returned in the Access-Accept packet in response to a Diameter Query request. It is initiated through an extension point script or through the Rule and Policy Engine by setting it to a new environment variable named Query-Service.

3GPPAuthorization

Select this service to enable 3GPP authorization of subscribers. For more information about 3GPP authorization, see Chapter 20, “Wireless Support.”

3GPP-Reverse-Authorization

Select this service to enable 3GPP reverse authorization of subscribers. For more information about 3GPP reverse authorization, see Chapter 20, “Wireless Support.”

Table 3-11 lists and describes the fields in the Services Details page. The fields listed below are the entire list of all the available types. The fields are displayed based on the type selected.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Name</td>
<td>Required; must be unique in the Services list.</td>
</tr>
<tr>
<td>Incoming Script</td>
<td>Optional; name of script to run when the service starts.</td>
</tr>
<tr>
<td>Type</td>
<td>Required; must set it to a valid Prime Access Registrar service.</td>
</tr>
<tr>
<td>Outgoing Script</td>
<td>Name of script to run when the service ends.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the service.</td>
</tr>
<tr>
<td>Outage Script</td>
<td>Optional; if you set this property to the name of a script, Cisco Prime Access Registrar runs it when an outage occurs. This property allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
</tbody>
</table>
### Table 3-11 Simple Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outage Policy</td>
<td>Required; the default is DropPacket. This property defines how Cisco Prime Access Registrar handles requests if all servers listed in the RemoteServers properties are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: AcceptAll, DropPacket, or RejectAll.</td>
</tr>
<tr>
<td></td>
<td>The following properties appear for the job type rex.</td>
</tr>
<tr>
<td>Filename</td>
<td>Required; must be either a relative or an absolute path to the shared library containing the Service. When the pathname is relative, it must be relative to $INSTALL/Scripts/RADIUS/rex.</td>
</tr>
<tr>
<td>EntryPoint</td>
<td>Required; must be set to the function’s global symbol.</td>
</tr>
<tr>
<td>InitEntryPoint</td>
<td>Required; must be the name of the global symbol Cisco Prime Access Registrar should call when it initializes the shared library and just before it unloads the shared library. A rex service must have an InitEntryPoint even if the service only returns REX_OK.</td>
</tr>
<tr>
<td>InitEntryPointArgs</td>
<td>Optional; when set, it provides the arguments to be passed to the InitEntryPoint in the environmental variable Arguments.</td>
</tr>
<tr>
<td></td>
<td>The following properties appear for the job type file.</td>
</tr>
<tr>
<td>FilenamePrefix</td>
<td>Required; a string that specifies where Cisco Prime Access Registrar writes the account records. It must be either a relative or absolute path. When you specify a relative path, it must be relative to the $INSTALL/logs directory. When you specify an absolute path, the server must be able to reach it. The default is Accounting.</td>
</tr>
<tr>
<td>MaxFileAge</td>
<td>Optional; stored as a string, but is composed of two parts, a number and a units indicator (&lt;n&gt; &lt;units&gt;) in which the unit is one of: H, Hour, Hours, D, Day, Days, W, Week, Weeks. The default is one day.</td>
</tr>
<tr>
<td>RolloverSchedule</td>
<td>Indicates the exact time including the day of the month or day of the week, hour and minute to roll over the accounting log file.</td>
</tr>
<tr>
<td>MaxFileSize</td>
<td>Optional; stored as a string, but is composed of two parts, a number and a units indicator (&lt;n&gt; &lt;units&gt;) in which the unit is one of: K, kilobyte, or kilobytes, M, megabyte, or megabytes, or G, gigabyte, or gigabytes. The default is ten megabytes.</td>
</tr>
<tr>
<td>UseLocalTimeZone</td>
<td>When set to TRUE, indicates the accounting records' TimeStamp is in local time. When set to FALSE, the default, accounting records' TimeStamp is in GMT.</td>
</tr>
<tr>
<td>FileType</td>
<td>Choose log or csv to indicate the file type to export the accounting records to. If you choose log, the Prime Access Registrar server writes accounting messages to the accounting.log file in the /opt/CSCOar/logs directory. If you choose csv, the Prime Access Registrar server writes accounting messages to the accounting.csv file in the /opt/CSCOar/logs directory.</td>
</tr>
<tr>
<td>EnableRollOverIntelligence</td>
<td>Check the box to enable rollover intelligence for the accounting records based on the accounting service properties. For more information, see EnableRollOverIntelligence, page 7-5.</td>
</tr>
</tbody>
</table>
Table 3-11  Simple Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttributesToBeLogged</td>
<td>The selected list of attributes that must be logged. If the list is empty, the accounting file service logs all the attributes of the packet.</td>
</tr>
<tr>
<td>Delimiter</td>
<td>The delimiter to use in the accounting file. This field is available if you set the FileType as <strong>csv</strong>. Delimiters could be ‘;’, ‘,’ and ‘;’ and default value is ‘,’.</td>
</tr>
</tbody>
</table>

The following properties appear for the job type **trusted-id**.

UserService Required; name of service that can be used to authenticate.

SessionManager Required; select the required session manager from the available list.

The following properties appear for the job type **group**.

Result Rule When set to AND (the default), the response from the GroupService is positive if each of the services referenced return a positive result. The response is negative if any of the services reference return a negative result. When set to OR, the response from the GroupService is positive if any of the services referenced return a positive result. The response is negative if all the referenced services return a negative result. The settings parallel-AND or parallel-OR are similar to AND and OR settings, except that each referenced service processes requests simultaneously instead of asking each reference service sequentially to save processing time.

GroupServices Optional; use the GroupServices subdirectory to specify the subservices in an indexed list to provide specific ordering control of which services to apply first. Each subservice listed must be defined in the Services section of the RADIUS configuration and cannot be of type group, eap-leap, or eap-md5.

To navigate between the listed attributes, use the navigation option available adjacent to the list. See **Relocating Records** for more details.

The following properties appear for the job type **local**.

UserList Required; this object contains all of the individual UserLists, which in turn, contain the specific users stored within Prime Access Registrar. Cisco Prime Access Registrar references each specific UserList by name from a Service whose type is set to **local**.

When Cisco Prime Access Registrar receives a request, it directs it to a Service. When the Service has its type property set to **local**, the Service looks up the user’s entry in the specific UserList and authenticates and/or authorizes the user against that entry.

Enable Device Access Check the box to enable TACACS+ command authorization.

**Note**  Device Access Rules are applicable for TACACS+ command authorization. For more information, see **TACACS+ Support for AAA**, page 17-57.

Device Access Rule Select a device access rule and click **Add**. The selected access rule is displayed in the Device Access Rules list box.

Default Device Access Action Select the default action to perform on the commands for all the access rules in the authorization service. Options are **PermitAll** and **DenyAll**.
The following properties appear for the job type `java`.

- **Class name** Optional; set to the name of a class that implements the Extension interface.
- **InitializeArg** Optional; set to a string to be passed to the Initialize method if the class implements the optional ExtensionWithInitialization interface.

The following properties appear for the job type `wimax`.

- **HARKKey** Required; used as the base key to generate random HARKKey for all the HAs that are configured in Prime Access Registrar. By default, the value is `cisco112`. You can change this value.
- **WimaxAuthenticationService** Required; a valid EAP service which can be used for WiMAX authentication. By default, this value is none.
- **HARKLifeTime** Required; used as time (in minutes) to regenerate the HARKKeys based on its lifetime.
- **WimaxSessionManager** Required; set a valid session manager which has HA and HA Cache as resource managers. By default, this value is none.
- **WimaxQueryService** Required; set a valid RADIUS query service which is configured with WiMAX session manager. By default, this value is none.
- **WimaxPrepaidService** Optional; set a valid prepaid service to carry out the prepaid functionality of WiMAX. Otherwise this value is set to none.
- **AllowAAAToIncludeKeys** Optional; If this is set, the HAAA will include the hHA-RK-Key, hHA-RK-SPI and hHA-RK-Lifetime in the Access-Accept. Otherwise, those attributes will not be in the Access-Accept. By default this value is True.
- **RequiredMSK** Optional; If this is set, the MSK will be provided by the AAA server as a result of successful EAP-Authentication. By default, this value is False.

The following properties appear for the job type `radius-query`.

- **Attribute List tab**
  - **Attribute type** Select either RADIUS or VENDOR. If Vendor is selected, specify the vendor type from the drop-down list. Select the attributes from the available list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details.

- **Session Manager tab**
  - **Session Manager** Select the required session manager from the available list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details.

The following property appears for the job type `dyn-auth`.

- **Session Cache Query Service** Select the session cache query service to use for dynamic authorization.

The following properties appear for the job type `diameter-radius` or `radius-diameter`.

- **ProxyServiceName** Select the Diameter proxy service name.
- **DiameterApplicationID** Select the Diameter service application ID. This field appears only for radius-diameter service type.
<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UseFor3GPPReverseAuthorizationService</td>
<td>Check the box to enable 3GPP authorization service in the translation framework. This field appears only for radius-diameter service type.</td>
</tr>
<tr>
<td>PreRequestTranslationScript</td>
<td>Select the scripting point to be called on the original request packet.</td>
</tr>
<tr>
<td>PostRequestTranslationScript</td>
<td>Select the scripting point to be called on the translated request packet.</td>
</tr>
<tr>
<td>PreResponseTranslationScript</td>
<td>Select the scripting point to be called on the response packet.</td>
</tr>
<tr>
<td>PostResponseTranslationScript</td>
<td>Select the scripting point to be called on the translated response packet.</td>
</tr>
<tr>
<td>ResultCodeMappings</td>
<td>This tab allows you to map result codes.</td>
</tr>
<tr>
<td>RequestAVPMappings</td>
<td>This tab allows you to map request AVPs.</td>
</tr>
<tr>
<td>RequestAVPsToBeAdded</td>
<td>This tab allows you to map request AVPs to be added.</td>
</tr>
<tr>
<td>RequestEnvironmentMappings</td>
<td>This tab allows you to map request environment variables.</td>
</tr>
<tr>
<td>ResponseAVPMappings</td>
<td>This tab allows you to map response AVPs.</td>
</tr>
<tr>
<td>ResponseAVPsToBeAdded</td>
<td>This tab allows you to map response AVPs to be added.</td>
</tr>
<tr>
<td>ResponseEnvironmentMappings</td>
<td>This tab allows you to map response environment variables.</td>
</tr>
</tbody>
</table>

The following properties appear for the job type **diameter-query**.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UpdateSessionLastAccessTime</td>
<td>Check the box to update the timestamp when the Diameter session was last accessed or called.</td>
</tr>
</tbody>
</table>

**Attribute List tab**

<table>
<thead>
<tr>
<th>Attribute type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select either <strong>RADIUS</strong> or <strong>VENDOR</strong>. If Vendor is selected, specify the vendor type from the drop-down list. Select the attributes from the available list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details.</td>
<td></td>
</tr>
</tbody>
</table>

**Session Manager tab**

<table>
<thead>
<tr>
<th>Session Manager</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select the required session manager from the available list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details.</td>
<td></td>
</tr>
</tbody>
</table>

The following property appears for the job type **3gpp-authorization**.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>Required; select RADIUS or Diameter to indicate the protocol to use for 3GPP authorization.</td>
</tr>
<tr>
<td>TranslationService</td>
<td>Required if the protocol selected is RADIUS; translation service to use during 3GPP authorization.</td>
</tr>
<tr>
<td>DiameterProxyService</td>
<td>Required if the protocol selected in Diameter; diameter proxy service to use during 3GPP authorization.</td>
</tr>
</tbody>
</table>
Table 3-11 Simple Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TranslationService</td>
<td>Required; the translation service to use for 3GPP reverse authorization.</td>
</tr>
</tbody>
</table>

Adding Simple Service Details

To add new simple service details:

**Step 1** Choose **Configuration > Services > Simple**. The Services List(REX, FILE, LOCAL, GROUP, JAVA...) page is displayed.

**Step 2** Click **Add** to add new simple service details. The Services Details page is displayed.

**Step 3** Enter the required details.

**Step 4** Click **Submit** to save the specified details in the Services List(REX, FILE, LOCAL, GROUP, JAVA...) page. Otherwise click **Cancel** to return to the Services List(REX, FILE, LOCAL, GROUP, JAVA...) page without saving the details.

On successful creation of the simple service properties, the Services List(REX, FILE, LOCAL, GROUP, JAVA...) page is displayed else a respective error message is displayed.

ServiceWithRS

The RemoteServers directory lists one or more remote servers to process access requests. The servers must also be listed in order under /Radius/RemoteServers. The order of the RemoteServers list determines the sequence for directing access requests when MultipleServersPolicy is set to RoundRobin mode. The first server in the list receives all access requests when MultipleServersPolicy is set to Failover mode.

The RemoteServers object can be used to specify the properties of the remote servers to which Services proxy requests. RemoteServers are referenced by name from the RemoteServers list in either the RADIUS, LDAP or TACACS-UDP Services.

Table 3-12 lists and describes the fields in the Services Details page.
### Table 3-12  Remote Server Service Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service Name</td>
<td>Required; name of the remote server service</td>
</tr>
<tr>
<td>Incoming Script</td>
<td>Optional; name of script to run when the service starts</td>
</tr>
<tr>
<td>Type</td>
<td>Required; Remote service Type must be set to one of the following: ldap, ldap-accounting, odbc-accounting, odbc, oci-accounting, oci, prepaid, radius, radius-session, or m3ua.</td>
</tr>
<tr>
<td>Outgoing Script</td>
<td>Optional; name of script to run when the service ends.</td>
</tr>
<tr>
<td>Outage Script</td>
<td>Optional; if you set this property to the name of a script, Prime Access Registrar runs it when an outage occurs. This property allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
<tr>
<td>Outage Policy</td>
<td>The default is <strong>DropPacket</strong>. This property defines how Prime Access Registrar handles requests if all servers listed in the <strong>RemoteServers</strong> properties are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: <strong>AcceptAll</strong>, <strong>DropPacket</strong>, or <strong>RejectAll</strong>.</td>
</tr>
<tr>
<td>Description (optional)</td>
<td>Optional; description of the remote server service</td>
</tr>
<tr>
<td>MultipleServersPolicy</td>
<td>Required; must be set to either <strong>Failover</strong> or <strong>RoundRobin</strong>.</td>
</tr>
<tr>
<td>When you set it to <strong>Failover</strong>, Prime Access Registrar directs requests to the first server in the list until it determines the server is offline. At which time, Prime Access Registrar redirects all requests to the next server in the list until it finds a server that is online.</td>
<td></td>
</tr>
<tr>
<td>When you set it to <strong>RoundRobin</strong>, Prime Access Registrar directs each request to the next server in the RemoteServers list to share the resource load across all of the servers listed in the RemoteServers list.</td>
<td></td>
</tr>
<tr>
<td>RemoteServers</td>
<td>Select the required remote server from the available list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See <strong>Relocating Records</strong> for more details.</td>
</tr>
<tr>
<td>AuthorizationInfoLookUp</td>
<td>Applicable only for the m3ua service type. Choose one of the following from the drop-down list:</td>
</tr>
<tr>
<td>• MSISDN-IMSI—To fetch MSISDN in the request and send IMSI in the response to the HLR.</td>
<td></td>
</tr>
<tr>
<td>• IMSI-MSISDN—To fetch IMSI in the request and send MSISDN in the response to the HLR.</td>
<td></td>
</tr>
<tr>
<td>• MAP-RESTORE—To fetch the profile information of a subscriber from the HLR. For more information on configuring the M3UA service with Map Restore Data authorization, see <strong>Configuring M3UA Service with Map Restore Data Authorization</strong>, page 23-15.</td>
<td></td>
</tr>
<tr>
<td>MapVersion</td>
<td>Applicable only for the m3ua service type; select the map version that HLR supports.</td>
</tr>
</tbody>
</table>

### Device Access Rules

This section is applicable for TACACS+ command authorization and is available only for service types local-user, oci, odbc, and ldap. For more information on TACACS+ command authorization, see **TACACS+ Support for AAA**, page 17-57.
## Table 3-12 Remote Server Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Device Access</td>
<td>Check the box to enable TACACS+ command authorization.</td>
</tr>
<tr>
<td>Device Access Rule</td>
<td>Select a device access rule and click Add. The selected access rule is displayed in the Device Access Rules list box.</td>
</tr>
<tr>
<td>Default Device Access Action</td>
<td>Select the default action to perform on the commands for all the access rules in the authorization service. Options are PermitAll and DenyAll.</td>
</tr>
</tbody>
</table>

### Restore Data Mappings Section

<table>
<thead>
<tr>
<th>IMSI</th>
<th>IMSI received in the response from HLR.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naea-Preferred CI</td>
<td>North American Equal Access preferred Carrier ID List. A list of the preferred carrier identity codes that are subscribed to.</td>
</tr>
<tr>
<td>Roaming Restricted In Sgsn Due To Unsupported Feature</td>
<td>Indicates that a subscriber is not allowed to roam in the current Service GPRS Support Node (SGSN) or Cisco Mobility Management Entity (MME) area.</td>
</tr>
<tr>
<td>Network Access Mode</td>
<td>The Network Access Mode (NAM) defines if the subscriber is registered to get access to the CS (non-GPRS/EPS network), to the PS (GPRS/EPS) network, or to both networks. NAM describes the first level of the subscriber data pseudo-tree below the IMSIroot. It is permanent subscriber data stored in the HSS / HLR and the SGSN with the Gs interface option, and the MME with the SGs interface option.</td>
</tr>
<tr>
<td>LMU Indicator</td>
<td>Indicates the presence of an LMU.</td>
</tr>
<tr>
<td>IST Alert Timer</td>
<td>Indicates the IST alert timer value that must be used in the Mobile Switching Center (MSC) to inform the HLR about the call activities that the subscriber performs.</td>
</tr>
<tr>
<td>Super Charger Supported In HLR</td>
<td>Indicates whether super charger concept is supported in HLR.</td>
</tr>
<tr>
<td>CS Allocation Retention Priority</td>
<td>Allocation-retention priority for Circuit Switched (CS). This parameter specifies relative importance to compare with other bearers about allocation and retention of bearer.</td>
</tr>
<tr>
<td>ChargingCharacteristics</td>
<td>Subscribed charging characteristics.</td>
</tr>
<tr>
<td>Access Restriction Data</td>
<td>Allowed Recipient Access Table (RAT) according to subscription data.</td>
</tr>
<tr>
<td>UE Reachability Request Indicator</td>
<td>Indicates that the Home Subscriber Server (HSS) is awaiting a notification of user equipment (UE) reachability.</td>
</tr>
<tr>
<td>Category</td>
<td>Calling party category</td>
</tr>
<tr>
<td>LSA Information</td>
<td>These parameters refer to one or more localized service areas (LSAs) a subscriber may be a member of, together with the priority, the preferential access indicator, the active mode support indicator and active mode indication of each localized service area. The access right outside these localized service areas is also indicated.</td>
</tr>
</tbody>
</table>

### Subscriber Data
### Table 3-12  Remote Server Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSISDN</td>
<td>MSISDN value in the subscriber data.</td>
</tr>
<tr>
<td>Subscriber Status</td>
<td>Barring status of the subscriber, which could be Service Granted or Operator Determined Barring.</td>
</tr>
<tr>
<td>Roaming Restriction Due To Unsupported Feature</td>
<td>Indicates that the subscriber is not allowed to roam in the current MSC area.</td>
</tr>
<tr>
<td>Bearer Service List</td>
<td>List of extensible bearer services subscribed.</td>
</tr>
<tr>
<td></td>
<td>Configure the index value to fetch only the required bearer services.</td>
</tr>
<tr>
<td>TeleService List</td>
<td>List of extensible teleservices subscribed.</td>
</tr>
<tr>
<td></td>
<td>Configure the index value to fetch only the required teleservices.</td>
</tr>
<tr>
<td>Provisioned SS</td>
<td>List of supplementary services provisioned.</td>
</tr>
<tr>
<td></td>
<td>Configure the index value to fetch only the required supplementary services.</td>
</tr>
<tr>
<td>ODB-Data</td>
<td>Operator Determined Barring (ODB) general data and ODB Home Public Land Mobile Network (HPLMN) specific data.</td>
</tr>
<tr>
<td>Regional Subscription Data</td>
<td>List of regional subscription areas (zones) in which the subscriber is allowed to roam.</td>
</tr>
<tr>
<td></td>
<td>Configure the index value to fetch only the required zones.</td>
</tr>
<tr>
<td>VBS Subscription Data</td>
<td>List of Voice Broadcast Services (VBS) subscribed.</td>
</tr>
<tr>
<td></td>
<td>Configure the index value to fetch only the required VBS.</td>
</tr>
<tr>
<td>VGCS Subscription Data</td>
<td>List of Voice Group Call Services (VGCS) subscribed.</td>
</tr>
<tr>
<td></td>
<td>Configure the index value to fetch only the required VGCS.</td>
</tr>
<tr>
<td><strong>LCS Information</strong></td>
<td>Live Communication Server (LCS) related information for the subscriber.</td>
</tr>
<tr>
<td>GMLC-List</td>
<td>List of Gateway Mobile Location Centers (GMLCs) that are permitted to issue a call/session unrelated or call/session related MT-LR request.</td>
</tr>
<tr>
<td></td>
<td>Configure the index value to fetch only the required GMLCs.</td>
</tr>
<tr>
<td>LCS-Privacy Exception List</td>
<td>Classes of LCS client that are allowed to locate any target Mobile Station (MS).</td>
</tr>
<tr>
<td></td>
<td>Configure the index value to fetch only the required classes.</td>
</tr>
<tr>
<td>MOLR-List</td>
<td>Code and status of Mobile Originating Location Request (MO-LR) subscribed.</td>
</tr>
<tr>
<td></td>
<td>Configure the index value to fetch only the required requests.</td>
</tr>
<tr>
<td><strong>MC-SS-Info</strong></td>
<td>Parameters identifying Multicall (MC) supplementary services (SS) that are subscribed.</td>
</tr>
<tr>
<td>MC-SS-Code</td>
<td>Code of the MC SS.</td>
</tr>
<tr>
<td>MC-SS-Status</td>
<td>Status of the MC SS.</td>
</tr>
<tr>
<td>NbrSB</td>
<td>Maximum number of parallel bearers that may be used as defined by the user’s subscription.</td>
</tr>
<tr>
<td>NbrUser</td>
<td>Maximum number of parallel bearers that may be used as defined by the user at registration of the MC SS.</td>
</tr>
</tbody>
</table>
You can use the ServiceWithRS List page for the following:

- Filtering Records
- Adding Remote Server Service Details
- Editing Records
- Deleting Records

Adding Remote Server Service Details

To add new remote server service details:

**Step 1** Choose **Configuration > Services > ServiceWithRS**. The Services List (..with Remote Servers) page is displayed.

**Step 2** Click **Add** to add new remote server service details. The Services Details page is displayed.

**Step 3** Enter the required details.

**Step 4** Click **Submit** to save the specified details in the Services List (..with Remote Servers) page. Otherwise, click **Cancel** to return to the Services List (..with Remote Servers) List page without saving the details.

On successful creation of the properties, the Services List (..with Remote Servers) page is displayed else a respective error message is displayed.

### PEAP Service

Protected EAP (PEAP) is an authentication method designed to mitigate several weaknesses of EAP. PEAP leverages Industry standard authentication of the server using certificates TLS (RFC 2246) and creation of a secure session that can then be used to authenticate the client.
The PEAP protocol consists of two phases, an authentication handshake phase and a tunnel phase where another complete EAP authentication exchange takes place protected by the session keys negotiated by phase one. Prime Access Registrar supports the tunneling of other EAP methods within the PEAP phase two exchange.

Prime Access Registrar supports the two major existing variants of PEAP:

- **PEAP Version 0** (Microsoft PEAP)
- **PEAP Version 1** (Cisco Prime PEAP)

### PEAP Version 0

PEAP Version 0 also called as Microsoft PEAP is described in IETF drafts (draft-kamath-pppext-peapv0-00.txt and draft-josefsson-pppext-eap-tls-eap-02.txt). This version of PEAP uses either EAP-MSChapV2 or EAP-SIM as an authentication method. The testing method used for this version of PEAP is radclient.

### PEAP Version 1

PEAP Version 1 also called as Cisco Prime PEAP is described by IETF draft (draft-zhou-pppext-peapv1-00.txt). This version can use either EAP-GTC or EAP-SIM as an authentication method. The testing method used for this version of PEAP is radclient.

Table 3-13 lists and describes the fields in the PEAP Services Details page. The fields listed below are the entire list of all the available types. The fields are displayed based on the type selected.

<table>
<thead>
<tr>
<th><strong>Table 3-13 PEAP Service Properties</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fields</strong></td>
</tr>
<tr>
<td>Service Name</td>
</tr>
<tr>
<td>Incoming Script</td>
</tr>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Outgoing Script</td>
</tr>
<tr>
<td>Maximum Message Size</td>
</tr>
<tr>
<td>Server Certificate File</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Private Key Password</td>
</tr>
<tr>
<td>Server RSA Key File</td>
</tr>
</tbody>
</table>
Optional; The URL that Prime Access Registrar should use to retrieve the CRL. You can specify a URL that uses HTTP or LDAP.

The following is an example for an HTTP URL:

The following is an example for an LDAP URL:
ldap://209.165.200.225:388/CN=development-CA,CN=acs-westcoast2,CN=CDP,CN=Public Key Services,CN=Services,CN=Configuration,DC=cisco,DC=com

Table 3-13 PEAP Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRL Distribution URL</td>
<td>Optional; The URL that Prime Access Registrar should use to retrieve the CRL. You can specify a URL that uses HTTP or LDAP. The following is an example for an HTTP URL: <a href="http://crl.verisign.com/pca1.1.1.crl">http://crl.verisign.com/pca1.1.1.crl</a>. The following is an example for an LDAP URL: ldap://209.165.200.225:388/CN=development-CA,CN=acs-westcoast2,CN=CDP,CN=Public Key Services,CN=Services,CN=Configuration,DC=cisco,DC=com</td>
</tr>
<tr>
<td>CA Certificate File</td>
<td>Optional; the full pathname of the file containing trusted CA certificates used for client verification. The file can contain more than one certificate, but all certificates must be in PEM format. DER encoding is not allowed.</td>
</tr>
<tr>
<td>Certificate Verification Mode</td>
<td>Optional; specifies the type of verification used for client certificates. Must be set to one of RequireCertificate, None, or Optional. - RequireCertificate causes the server to request a client certificate and authentication fails if the client refuses to provide one. - None will not request a client certificate. Optional causes the server to request a client certificate but the client is allowed to refuse to provide one.</td>
</tr>
<tr>
<td>CA Certificate Path</td>
<td>Optional; the name of a directory containing trusted CA certificates (in PEM format) used for client verification. This parameter is optional, and if it is used there are some special preparations required for the directory it references. Each certificate file in this directory must contain exactly one certificate in PEM format. The server looks up the certificate files using the MD5 hash value of the certificate’s subject name as a key. The directory must therefore also contain a set of symbolic links each of which points to an actual certificate file. The name of each symbolic link is the hash of the subject name of the certificate. For example, if a certificate file name <em>ca-cert.pem</em> is located in the CA Certificate Path directory, and the MD5 hash of the subject name contained in <em>ca-cert.path.pem</em> is 1b966d93, then a symbolic link named 1b966d93 must point to the <em>ca-cert.pem</em> file. If there are subject name collisions such as multiple certificates with the same subject name, each link name must be indexed with a numeric extension as in 1b966d93.0 and 1b966d93.1.</td>
</tr>
<tr>
<td>Verification Depth</td>
<td>Optional; specifies the maximum length of the certificate chain used for client verification.</td>
</tr>
<tr>
<td>Enable Session Cache</td>
<td>Optional; specifies whether TLS session caching (fast reconnect) is enabled or not. Set to True to enable session caching; otherwise set to False.</td>
</tr>
<tr>
<td>Tunnel Service</td>
<td>Required; must be the name of an existing EAP-MSCHAPv2 or EAP-SIM service.</td>
</tr>
<tr>
<td>Authentication Timeout</td>
<td>Required; specifies time (in seconds) to wait before an authentication request times out; defaults to 120.</td>
</tr>
<tr>
<td>Description (optional)</td>
<td>Optional; description of the PEAP service.</td>
</tr>
</tbody>
</table>
You can use the PEAP Services List page for the following:

- Filtering Records
- Adding PEAP Service Details
- Editing Records
- Deleting Records

**Adding PEAP Service Details**

To add new PEAP service details:

1. Choose **Configuration > Services > PEAP**. The PEAP Services List page is displayed.
2. Click **Add** to add new PEAP service details. The PEAP Services Details page is displayed.
3. Specify the relevant PEAP service details.
4. Click **Submit** to save the specified details in the PEAP Services List page. Otherwise click **Cancel** to return to the PEAP Services List page without saving the details.

**Table 3-13 PEAP Service Properties (continued)**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session Timeout</td>
<td>Optional; if TLS session caching (fast reconnect) is enabled, SessionTimeout specifies the maximum lifetime of a TLS session. Expired sessions are removed from the cache and will require a subsequent full authentication. SessionTimeout is specified as a string consisting of pairs of numbers and units, where units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks, as in the following: <strong>Set SessionTimeout “1 Hour 45 Minutes”</strong></td>
</tr>
<tr>
<td>Use ECC Certificates</td>
<td>Check this box, to use the ECC, RSA, or combination of both the certificates for certificate based verification. When this field is disabled, only RSA is used for certificate based verification. The default location to fetch the certificate file is /cisco-ar/pki.</td>
</tr>
<tr>
<td>Enable Auto Chaining</td>
<td>Check this box to enable grouping of relevant certificates.</td>
</tr>
<tr>
<td>Enable WPS</td>
<td>Optional; When set to TRUE, enables Windows Provisioning Service (WPS) and provides two other properties, MasterURL and WPSGuestUserProfile. The default value is FALSE.</td>
</tr>
<tr>
<td>Master URL</td>
<td>Optional; when using WPS, specifies the URL of the provisioning server which is modified with the appropriate fragment and sent to the client.</td>
</tr>
<tr>
<td>WPS Guest User Profile</td>
<td>Optional; when using WPS, specifies a profile to be used as a guest user profile; must be a valid profile under /Radius/Profiles. This profile is used for guests and users whose account has expired. This profile normally contains attributes denoting the VLAN-id of the guest network (which has the provisioning server alone) and might contain IP-Filters that would restrict the access of the guest (to only the provisioning server).</td>
</tr>
</tbody>
</table>
On successful creation of the PEAP service properties, the PEAP Services List page is displayed else a respective error message is displayed.

**EAP Service**

Prime Access Registrar supports the Extensible Authentication Protocol (EAP) to provide a common protocol for differing authentication mechanisms. It provides dynamic selection of the authentication mechanism at the time of authentication based on information transmitted in the Access-Request.

Prime Access Registrar supports the following EAP authentication methods:

- EAP-AKA
- EAP-AKA-Prime
- EAP-FAST
- EAP-GTC
- EAP-LEAP
- EAP-MD5
- EAP-Negotiate
- EAP-MSChapV2
- EAP-SIM
- EAP-Transport Level Security (TLS)
- EAP-TTLS

**EAP-AKA**

Authentication and Key Agreement (AKA) is an EAP mechanism for authentication and session key distribution. It is used in the 3rd generation mobile networks Universal Mobile Telecommunications System (UMTS) and CDMA2000. AKA is based on symmetric keys, and typically runs in a UMTS Subscriber Identity Module (USIM), or a (Removable) User Identity Module ((R) UIM), similar to a smart card. EAP-AKA (Extensible Authentication Protocol Method for UMTS Authentication and Key Agreement) includes optional identity privacy support, optional result indications, and an optional fast reauthentication procedure. The EAP-AKA authentication service is extended to generate a Diameter message Multimedia-Authentication-Request (MAR), with the subscriber identity (IMSI), to the Home Subscriber Server (HSS) when it requires the authentication vectors. The HSS sends a Diameter Multimedia-Authentication-Answer (MAA) back containing the number of quintuplets.

**EAP-AKA-Prime**

EAP-AKA-Prime (EAP-AKA’) is an EAP authentication method, with a small revision to the existing EAP-AKA method. EAP- AKA’ has a new key derivation function, which binds the keys derived within the method to the name of the access network. This limits the effects of compromised access network nodes and keys. EAP-AKA’ supports SHA-256 instead of SHA-1.
EAP-FAST

EAP-FAST is an authentication method which uses the EAP-MSChapV2 method for credential provisioning and EAP-GTC for authentication. Credential provisioning typically occurs only during the client’s initial EAP-FAST authentication. Subsequent authentications rely on the provisioned credential and will usually omit the provisioning step.

This authentication protocol is designed to address the performance shortcomings of prior TLS-based EAP methods while retaining features such as identity privacy and support for password-based protocols. The EAP-FAST protocol is described by the IETF draft (draft-cam-winget-eap-fast-00.txt).

EAP-GTC

This method defined in RFC 2284, is used for transmitting a username and password to an authentication server.

*Note*

It should not be used except as an authentication method for PEAP Version 1 because the password is not protected.

EAP-LEAP

The new AAA Cisco-proprietary protocol called Light Extensible Authentication Protocol (LEAP) supported by Prime Access Registrar, is a proprietary Cisco authentication protocol designed for use in IEEE 802.11 wireless local area network (WLAN) environments. Important features of LEAP include:

- Mutual authentication between the network infrastructure and the user
- Secure derivation of random, user-specific cryptographic session keys
- Compatibility with existing and widespread network authentication mechanisms (e.g., RADIUS)

*Note*

Prime Access Registrar supports a subset of EAP to support LEAP. This is not a general implementation of EAP for Prime Access Registrar.

The Cisco-Wireless or LEAP is an EAP authentication mechanism where the user password is hashed based on an MD4 algorithm.

EAP-MD5

This is another EAP authentication exchange. In EAP-MD5 there is a CHAP-like exchange and the password is hashed by a challenge from both client and server to verify the password. On successful verification, the connection proceeds, although the connection is periodically rechallenged (per RFC 1994).

EAP-Negotiate

This is a special service used to select at runtime the EAP service to be used to authenticate the client. It is configured with a list of candidate EAP services that represent the allowable authentication methods in preference order.

EAP-Negotiate is useful when the client population has deployed a mix of different EAP methods that must be simultaneously supported by Prime Access Registrar. EAP-Negotiate solves the problem of distinguishing client requirement by using the method negotiation feature of the EAP protocol.
EAP-MSChapV2
EAP-MSChapv2 encapsulates the MSChapV2 protocol (specified by RFC 2759) and can be used either as an independent authentication mechanism or as an inner method for PEAP Version 0 (recommended). This is based on draft-kamath-pppext-eap-mschapv2-00.txt, an informational IETF draft document.

EAP-SIM
An access point uses the Prime Access Registrar RADIUS server to perform EAP-SIM authentication of mobile clients. Prime Access Registrar must obtain authentication information from the HLR. Prime Access Registrar contacts the MAP gateway that performs the MAP protocol over SS7 to the HLR, or alternately it can contact the HLR (through STP in some cases) using the SIGTRAN-M3UA interface. The EAP-SIM authentication service is extended to generate a Diameter message Multimedia-Authentication-Request (MAR), with the subscriber identity (IMSI), to the HSS when it requires the authentication vectors. The HSS sends a Diameter Multimedia-Authentication-Answer (MAA) back containing the number of triplets.

EAP-Transport Level Security (TLS)
This is an authentication method (described in RFC 2716) which leverages TLS, described in RFC 2246, to achieve certificate-based authentication of the server and the client (optionally). It provides many of the same benefits as PEAP but differs in the lack of support for legacy authentication methods.

EAP-TTLS
The Extensible Authentication Protocol Tunneled TLS (EAP-TTLS) is an EAP protocol that extends EAP-TLS. EAP-TTLS extends the authentication negotiation EAP-TLS by using the secure connection established by the TLS handshake to exchange additional information between client and server. It leverages TLS (RFC 2246) to achieve certificate-based authentication of the server (and optionally the client) and creation of a secure session that can then be used to authenticate the client using a legacy mechanism.

EAP-TTLS is a two-phase protocol. Phase 1 conducts a complete TLS session and derives the session keys used in Phase 2 to securely tunnel attributes between the server and the client. The attributes tunneled during Phase 2 can be used to perform additional authentication(s) via a number of different mechanisms.

The authentication mechanisms used during Phase 2 include PAP, CHAP, MS-CHAP, MS-CHAPv2, and EAP. If the mechanism is EAP, then several different EAP methods are possible.

Table 3-14 lists and describes the fields in the EAP Services Details page. The fields listed below are the entire list of all the available types. The fields are displayed based on the type selected.

<table>
<thead>
<tr>
<th>Table 3-14</th>
<th>EAP Service Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fields</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>Service Name</td>
<td>Required; service name</td>
</tr>
<tr>
<td>Incoming Script</td>
<td>Optional script Prime Access Registrar server runs when it receives a request from a client.</td>
</tr>
<tr>
<td>Type</td>
<td>Required; must set it to a valid Prime Access Registrar service</td>
</tr>
<tr>
<td>Outgoing Script</td>
<td>Optional script Prime Access Registrar server runs before it sends a response to a client</td>
</tr>
<tr>
<td>Description (optional)</td>
<td>Optional; description of the PEAP service.</td>
</tr>
<tr>
<td>Authentication Timeout</td>
<td>Mandatory; specifies time (in seconds) to wait before an authentication request times out; defaults to 120.</td>
</tr>
</tbody>
</table>
### Table 3-14 EAP Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserService</td>
<td>Required; name of service that can be used to authenticate using cleartext passwords.</td>
</tr>
<tr>
<td>ServiceList</td>
<td>List of preconfigured EAP authentication services. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details.</td>
</tr>
<tr>
<td>Maximum Message Size</td>
<td>Required; indicates the maximum length in bytes that a PEAP message can have before it is fragmented.</td>
</tr>
<tr>
<td>Server Certificate File</td>
<td>Required; the full pathname of the file containing the server’s certificate or certificate chain used during the TLS exchange. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The two valid encoding prefixes are PEM and DER. If an encoding prefix is not present, the file is assumed to be in PEM format.</td>
</tr>
<tr>
<td>Private Key Password</td>
<td>Required; the password used to protect the server’s private key.</td>
</tr>
<tr>
<td>Server Key File</td>
<td>Required; the full path name of the file containing the server’s RSA private key. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The two valid encoding prefixes are “PEM” and “DER”. If an encoding prefix is not present, the file is assumed to be in PEM format.</td>
</tr>
<tr>
<td>CRL Distribution URL</td>
<td>Optional; enter the URL that Prime Access Registrar should use to retrieve the CRL. You can specify a URL that uses HTTP or LDAP. The following is an example for an HTTP URL: <a href="http://crl.verisign.com/pca1.1.1.crl">http://crl.verisign.com/pca1.1.1.crl</a>. The following is an example for an LDAP URL: ldap://209.165.200.225:388/CN=development-CA,CN=acs-west coast2,CN=CDP,CN=Public Key Services,CN=Services,CN=Configuration,DC=cisco,DC=com</td>
</tr>
<tr>
<td>CA Certificate File</td>
<td>Optional; the full path name of the file containing trusted CA certificates used for client verification. The file can contain more than one certificate, but all certificates must be in PEM format. DER encoding is not allowed.</td>
</tr>
<tr>
<td>Certificate Verification Mode</td>
<td>The value is set to optional by default. If set to RequireCertificate, the client certificate will always be verified. If set to optional, client certificate verification happens optionally.</td>
</tr>
</tbody>
</table>
### Table 3-14  EAP Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
</table>
| CA Certificate Path     | The name of a directory containing trusted CA certificates (in PEM format) used for client verification. This parameter is optional and if it is used there are some special preparations required for the directory it references.  
Each certificate file in this directory must contain exactly one certificate in PEM format. The server looks up the certificate files using the MD5 hash value of the certificate's subject name as a key.  
The directory must therefore also contain a set of symbolic links each of which points to an actual certificate file. The name of each symbolic link is the hash of the subject name of the certificate.  
For example, if a certificate file named `ca-cert.pem` is located in the CACertificatePath directory, and the MD5 hash of the subject name contained in `ca-cert.path.pem` is 1b96dd93, then a symbolic link named 1b96dd93 must point to `ca-cert.pem`.  
If there are subject name collisions such as multiple certificates with the same subject name, each link name must be indexed with a numeric extension as in 1b96dd93.0 and 1b96dd93.1. |
| Verification Depth      | Optional; specifies the maximum length of the certificate chain used for client verification.                                                                                                                                                               |
| Enable Session Cache    | Optional; specifies whether TLS session caching (fast reconnect) is enabled or not. Set to True to enable session caching; otherwise set to False.                                                                                                             |
| Session Timeout         | Required; if TLS session caching (fast reconnect) is enabled, SessionTimeout specifies the maximum lifetime of a TLS session. Expired sessions are removed from the cache and will require a subsequent full authentication.  
SessionTimeout is specified as a string consisting of pairs of numbers and units, where units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks, as in the following:  
Set SessionTimeout “1 Hour 45 Minutes”                                                                                                                                 |
| UseECCCertificate       | Determines the applicability of the authentication mechanism in SmartGrid Solutions.  
When you check this check box, it can use the ECC, RSA, or combination of both certificate for certificate based verification.  
When you uncheck this check box, it can only use the RSA certificate for certificate based verification. The default location to fetch the certificate file is `/cisco-ar/pki`. |
| EnableAutoChaining      | When set to TRUE, Prime Access Registrar sends its server certificate chain (Server-Cert -> IntermediateCA -> RootCA) while presenting the server certificate to the client for server side authentication. When set to FALSE, Prime Access Registrar sends only the server certificate (Server-Cert) to the client.                           |
### Table 3-14  EAP Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authentication Service</td>
<td>Specifies the name of the EAP-GTC service used for authentication. The named service must have the UseLabels parameter set to True.</td>
</tr>
<tr>
<td>User Prompt</td>
<td>Optional string the client might display to the user; default is Enter password:” Use the <code>set</code> command to change the prompt, as in the following: set UserPrompt “Admin Password:”</td>
</tr>
<tr>
<td>UseLabels</td>
<td>Required; must be set to TRUE for EAP-FAST authentication and set to FALSE for PEAP authentication. Set to FALSE by default.</td>
</tr>
<tr>
<td>SystemID</td>
<td>Optional; string that identifies the sender of the MSChapV2 challenge message.</td>
</tr>
<tr>
<td>IsWindows7Client</td>
<td>Optional; must be set to TRUE for EAP-MSChapV2 authentication. Set to FALSE by default.</td>
</tr>
<tr>
<td>Authority Identifier</td>
<td>Required; a string that uniquely identifies the credential (PAC) issuer. The client uses this value to select the correct PAC to use with a particular server from the set of PACs it might have stored locally.</td>
</tr>
<tr>
<td>Authority Information</td>
<td>Required; a string that provides a descriptive text for this credential issuer. The value can be displayed to the client for identification purposes and might contain the enterprise or server names.</td>
</tr>
<tr>
<td>Credential Life Time</td>
<td>Optional; specifies the maximum lifetime of a Protected Access Credential (PAC). Clients that successfully authenticate with an expired PAC will be reprovisioned with a new PAC. CredentialLifetime is specified as a string consisting of pairs of numbers and units, where units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks. Credentials that never expire should be specified as Forever.</td>
</tr>
<tr>
<td>Provision Service</td>
<td>Required; specifies the name of the EAP-MSChapV2 service used for provisioning.</td>
</tr>
<tr>
<td>Provision Mode</td>
<td>Required; specifies the TLS mode used for provisioning. Clients only support the default Anonymous mode.</td>
</tr>
<tr>
<td>Always Authenticate</td>
<td>Optional; indicates whether provisioning should always automatically rollover into authentication without relying on a separate session. Most environments, particularly wireless, will perform better when this parameter is set to True, the default value.</td>
</tr>
<tr>
<td>SubscriberDBLookup</td>
<td>Specifies the type of communication with the HLR/HSS server. Based on the type selected, the communication happens with the HLR/HSS server using the diameter Wx interface, MAP protocol, or SIGTRAN-M3UA protocol. This field is displayed when you select the eap-sim option in the Type field.</td>
</tr>
</tbody>
</table>
Table 3-14  EAP Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subscriber_DBLookup</td>
<td>Specifies the type of communication with the HLR/HSS server. Based on the type selected, the communication happens with the HLR/HSS server using the diameter Wx interface, SIGTRAN protocol, or SIGTRAN-M3UA protocol. This field is displayed when you select the eap-sim, eap-aka, or eap-aka’ option in the Type field.</td>
</tr>
<tr>
<td>DestinationRealm</td>
<td>Required. Destination realm to send Diameter packets to the remote server. The role of the remote server should be Relay.</td>
</tr>
<tr>
<td>PreRequestTranslationScript</td>
<td>Optional. Prime Access Registrar server runs before sending the request to the Diameter remote server. The script can modify the RADIUS packet dictionaries.</td>
</tr>
<tr>
<td>PostRequestTranslationScript</td>
<td>Optional. Prime Access Registrar server runs before sending the request to the Diameter remote server. The script can modify the Diameter packet dictionaries.</td>
</tr>
<tr>
<td>PreResponseTranslationScript</td>
<td>Optional. Prime Access Registrar server runs after receiving the response from the Diameter remote server. The script can modify the Diameter packet dictionaries.</td>
</tr>
<tr>
<td>PostResponseTranslationScript</td>
<td>Optional. Prime Access Registrar server runs after receiving the response from the Diameter remote server. The script can modify the RADIUS packet dictionaries.</td>
</tr>
<tr>
<td>FetchAuthorizationInfo</td>
<td>When you check this check box, it fetches MSISDN from HLR.</td>
</tr>
</tbody>
</table>

**General tab**

The details in the tab is displayed based on the eap-sim, eap-aka, or eap-aka-prime option you select in the Type field.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MultipleServersPolicy</td>
<td>Required. Must be set to either Failover or RoundRobin. When set to Failover, Prime Access Registrar directs requests to the first server in the list until it determines the server is offline. At that time, Prime Access Registrar redirects all requests to the next server in the list until it finds a server that is online. When set to RoundRobin, Prime Access Registrar directs each request to the next server in the RemoteServers list to share the resource load across all of the servers listed in the RemoteServers list.</td>
</tr>
<tr>
<td>NumberOfTriplets</td>
<td>Required; number of triplets (1, 2, or 3) to use for authentication; default is 2.</td>
</tr>
<tr>
<td>PseudonymSecret</td>
<td>Required; the secret string that is used as the basis for protecting identities when identity privacy is enabled. This should be at least 16 characters long and have a value that is impossible for an outsider to guess. The default value is secret. This field is not available if EnableRollingPseudonymSecret field is checked. <strong>Note</strong> It is very important to change PseudonymSecret from its default value to a more secure value when identity privacy is enabled for the first time.</td>
</tr>
</tbody>
</table>
### Table 3-14  EAP Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PseudonymRenewtime</td>
<td>Required; specifies the maximum age a pseudonym can have before it is renewed. When the server receives a valid pseudonym that is older than this, it generates a new pseudonym for that subscriber. The value is specified as a string consisting of pairs of numbers and units, where the units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks. The default value is “24 Hours”. Examples are: “8 Hours”, “10 Hours 30 Minutes”, “5 D 6 H 10 M”</td>
</tr>
<tr>
<td>PseudonymLifetime</td>
<td>Required; specifies the maximum age a pseudonym can have before it is rejected by the server, forcing the subscriber to authenticate using it’s permanent identity. The value is specified as a string consisting of pairs of numbers and units, where the units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks. It can also be Forever, in which case, pseudonyms do not have a maximum age. The default value is “Forever”. Examples are: “Forever”, “3 Days 12 Hours 15 Minutes”, “52 Weeks”</td>
</tr>
<tr>
<td>NotificationService</td>
<td>(Optional); Notification service is an authorization service and is used to send a notification code to the client in case of an authorization failure. For more information about the Notification-Code variable, see Appendix B, “Environment Dictionary.” This can be any of the services configured under /radius/services/ except eap services, accounting services, radius-session, radius-query, and diameter.</td>
</tr>
<tr>
<td>ReauthenticationTimeout</td>
<td>Required; specifies the time in seconds that reauthentication identities are cached by the server. Subscribers that attempt to reauthenticate using identities that are older than this value will be forced to use full authentication instead. The default value is 3600 (one hour).</td>
</tr>
<tr>
<td>EnableReauthentication</td>
<td>Optional; when True, the fast reauthentication option is enabled. The default value is False.</td>
</tr>
<tr>
<td>UseOutagePolicyforReauth</td>
<td>Default value is FALSE. When set to TRUE, Prime Access Registrar drops or rejects reauthentication requests as per outage policy when the remote server is down. This can be processed only when there is at least one failed full authentication before proceeding with reauthentication.</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required for EAP-SIM, EAP-AKA, and EAP-AKA’ services; the default is DropPacket. This property defines how Prime Access Registrar handles requests if all servers listed in the RemoteServers tab are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: AcceptAll, DropPacket, or RejectAll.</td>
</tr>
</tbody>
</table>
### Table 3-14 | EAP Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UseProtectedResults</td>
<td>Optional; enables or disables the use of protected results messages. Results messages indicate the state of the authentication but are cryptographically protected.</td>
</tr>
<tr>
<td>ReauthenticationRealm</td>
<td>Optional; realm to use for reauthentication.</td>
</tr>
<tr>
<td>MaximumReauthentications</td>
<td>Required; specifies the maximum number of times a reauthentication identity might be reused before it must be renewed. The default value is 16.</td>
</tr>
<tr>
<td>TripletCacheTimeout</td>
<td>Required for eap-sim service; time in seconds an entry remains in the triplet cache. A zero (0) indicates that triplets are not cached. The maximum is 28 days; the default is 0 (no caching).</td>
</tr>
<tr>
<td>QuintetCacheTimeout</td>
<td>Required for eap-aka or eap-aka’ service; time in seconds an entry remains in the quintet cache. A zero (0) indicates that quintets are not cached. The maximum is 28 days; the default is 0 (no caching).</td>
</tr>
<tr>
<td>QuintetGenerationScript</td>
<td>Available for eap-aka or eap-aka’ service; script required for quintet generation.</td>
</tr>
<tr>
<td>Authentication Timeout</td>
<td>Required; time in seconds to wait for authentication to complete. The default is 2 minutes; range is 10 seconds to 10 minutes.</td>
</tr>
<tr>
<td>UseSimDemoTriplets</td>
<td>Optional; set to TRUE to enable the use of demo triplets. This must be disabled for release builds.</td>
</tr>
<tr>
<td>AlwaysRequestIdentity</td>
<td>Optional; when True, enables the server to obtain the subscriber’s identity via EAP/SIM messages instead of relying on the EAP messages alone. This might be useful in cases where intermediate software layers can modify the identity field of the EAP-Response/Identity message. The default value is False.</td>
</tr>
<tr>
<td>EnableIdentityPrivacy</td>
<td>Optional; when True, the identity privacy feature is enabled. The default value is False.</td>
</tr>
<tr>
<td>Generate3GPPCompliantPseudonym</td>
<td>Optional; the value is set to False by default. If set to TRUE then Prime Access Registrar generates a 12 octet 3GPP compliant pseudonym identity. The Pseudonym username identities are used to protect the privacy of subscriber identities.</td>
</tr>
<tr>
<td>SendReAuthIDInAccept</td>
<td>Optional; the value is set to False by default. When set to True, Prime Access Registrar sends SN-Fast-ReAuth-UserName (Starent VSA) in access-accept message.</td>
</tr>
<tr>
<td>Outage Script</td>
<td>Optional; if you set this property to the name of a script, Prime Access Registrar runs it when an outage occurs. This property allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
<tr>
<td>NetworkName</td>
<td>Required for eap-aka-prime service type. Name of the access network for which the authentication is performed. This attribute is captured to ensure that the peer and the server know the name of the access network for performing the EAP authentication.</td>
</tr>
<tr>
<td>MapVersion</td>
<td>Required for SIGTRAN-M3UA remote server; select the map version HLR supports.</td>
</tr>
</tbody>
</table>
You can use the EAP Services List page for the following:

- Filtering Records
- Adding EAP Service Details
- Editing Records
- Deleting Records

**Adding EAP Service Details**

To add new EAP service details:

1. Choose **Configuration > Services > EAP**. The EAP Services List page is displayed.
2. Click **Add** to add new EAP service details. The EAP Services Details page is displayed.
3. Enter the relevant details.
4. Click **Submit** to save the specified details in the EAP Services List page. Otherwise click **Cancel** to return to the EAP Services List page without saving the details.

On successful creation of the EAP Service properties, the EAP Services List page is displayed else a respective error message is displayed.

**Diameter Service**

Proxy agents assist in routing Diameter messages using the Diameter routing table. Diameter proxy service works in tandem with the rule policy engine to perform the routing for multiple realms or applications. The following are the multiple peer policies supported by the proxy service:

- Round Robin
- FailOver
- IMSI Range Based.

Table 3-15 lists and describes the fields in the Diameter-Services page. The fields listed below are the entire list of all the available roles. The fields are displayed based on the role selected.

**Table 3-15  Diameter Service Properties**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; name of the Diameter server.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the Diameter server.</td>
</tr>
<tr>
<td>Realm</td>
<td>Required; realm of the route. Must be unique for a route table.</td>
</tr>
<tr>
<td>Role</td>
<td>Required; specifies the role that the Diameter entity will play in resolving messages matching the realm. The role can be any one of the following: Relay - Application acting as a Relay Agent. Redirect - Application acting as a Redirect Agent. Proxy - Application acting as a Proxy Agent. When the role is set to Proxy, the IncomingScript and OutgoingScript points are enabled. Local - Application processes the requests locally. When the role is set to Local, the Authentication Service and Accounting Service are enabled. By default, the Proxy option is selected. However, you can select another option from the drop-down list.</td>
</tr>
<tr>
<td>Incoming Script</td>
<td>Optional; enabled when role is set to Proxy or Local. When set, must be the name of a known incoming script. Prime Access Registrar runs the IncomingScript before proxying the Diameter packet to the remote Diameter server.</td>
</tr>
<tr>
<td>Outgoing Script</td>
<td>Optional; enabled when role is set to Proxy or Local. When set, must be the name of a known outgoing script. Prime Access Registrar runs the OutgoingScript after it receives the response from the remote Diameter server.</td>
</tr>
<tr>
<td>Authentication Service</td>
<td>Required; used when service is configured to process the Diameter requests locally. Set to valid service of type (local/ldap/odbc) to authenticate the user. This field is displayed when you select the role type as ‘Local’ in the Role field.</td>
</tr>
<tr>
<td>Accounting Service</td>
<td>Required; used when service is configured to process the accounting requests locally. Set to valid accounting service of type (file/odbc-accounting) to write the accounting records. This field is displayed when you select the role type as ‘Local’ in the Role field.</td>
</tr>
<tr>
<td>Type</td>
<td>Required; specifies the service type. The service type ‘Diameter’ is automatically displayed in this field.</td>
</tr>
</tbody>
</table>

**PEER Statements**

This is displayed when you select the ‘Local’, ‘Relay’, or ‘Redirect’ option in the Role field.
### Table 3-15  Diameter Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; name of the peer.</td>
</tr>
<tr>
<td>Host Name</td>
<td>Required; the hostname or IP address of the peer. The hostname must exist in the client list for the route to be active.</td>
</tr>
<tr>
<td>Metric</td>
<td>Required; metric value for the peer entry. The higher the value the lower the preference. The highest value of preference is 0.</td>
</tr>
<tr>
<td>VendorSpecific</td>
<td>Required; the default is FALSE. If set to FALSE, the application is ordinary application and user is prompted to enter the ApplicationID. If set to TRUE, the application is a VendorSpecific Application. User is prompted to enter VendorSpecificApplicationID and VendorID.</td>
</tr>
<tr>
<td>VendorID</td>
<td>Required; specifies the VendorID for the application. Example: DIAMETER 3GPP Cx APPLICATION VendorSpecificApplicationID 16777216 VendorID 10415</td>
</tr>
<tr>
<td>VendorSpecificApplicationID</td>
<td>Required; specifies the integer value for the vendor specific application.</td>
</tr>
<tr>
<td>ApplicationID</td>
<td>Required; application used in the route. The application Id should be available in /Advanced/Diameter/Applications.</td>
</tr>
</tbody>
</table>

### Applications
This is displayed when you select the ‘Proxy’ option in the Role field.

<table>
<thead>
<tr>
<th>Name</th>
<th>Required; name of the application.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The description of the application.</td>
</tr>
<tr>
<td>ApplicationID</td>
<td>Required; specifies the unique integer value for the application. It represents the application id of the Application used for load balancing the Diameter messages.</td>
</tr>
<tr>
<td>EnableSticky</td>
<td>Required; default is FALSE. If set to True, the sticky entries for load balancing is enabled and the user is prompted to enter the values for StickySessionKey, StickyCreationCmdList, and StickyDeletionCmdList.</td>
</tr>
<tr>
<td>MultiplePeersPolicy</td>
<td>Required; must be set to RoundRobin, FailOver, or IMSIRangeBased. Policy used by the Prime Access Registrar server to load balance the peers.</td>
</tr>
</tbody>
</table>
StickySessionKey Required; used as the sticky key for mapping the sticky sessions. Set the value to a valid attribute-value pair (AVP) in order to use the sticky key for maintaining Diameter sessions. This ensures that Prime Access Registrar maps the request to the same server for all the subsequent messages using the sticky key. For example, set StickyAVP “Session-Id”.

When the Prime Access Registrar server receives the CCR-I request, Prime Access Registrar extracts the Session-Id from the request packet, maps the Session to the peer configured in the list, and forwards the request to the chosen peer. Prime Access Registrar chooses the same peer for all the subsequent messages (CCR-Update/CCR-Terminate) with same Session-Id.

StickyCreationCmdList Required; specifies the command list to create the sticky entries. Specify the list of ‘|’ separated command code, AVP name, and its value to create the sticky sessions.

The following is the StickyCreationCmdList format:

<commandcode1>::<AVPName1=Value1> || <commandcode2>::<AVPName2=Value2> || <commandcode3>

For example, if the sticky session entries need to be created based on command code ‘265’ or based on command code ‘271’ with Accounting-Record-Type value as 2, use the format below:

Set StickyCreationCmdList "265||271::Accounting-Record-Type=2"

StickyDeletionCmdList Required; specifies the command list to delete the sticky entries. Specify the list of ‘|’ separated command code, AVP name, and its value to delete the sticky sessions.

The following is the StickyDeletionCmdList format:

<commandcode1>::<AVPName1=Value1> || <commandcode2>::<AVPName2=Value2> || <commandcode3>

For example, if the sticky session entries need to be deleted based on command code ‘271’ with Accounting-Record-Type value as 4, use the format below:

Set StickyDeletionCmdList "271::Accounting-Record-Type=4"

PEER Definitions Proxy

Name Required; name of the peer.

Host Name Required; hostname or IP address of the peer. The HostName must exist in the client list for the route to be active.

Metric Required; metric value for this peer entry. The higher the value the lower the preference. The highest value of preference is 0.

Table 3-15 Diameter Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StickySessionKey</td>
<td>Required; used as the sticky key for mapping the sticky sessions. Set the value to a valid AVP</td>
</tr>
<tr>
<td></td>
<td>in order to use the sticky key for Diameter sessions. This ensures that Prime Access Registrar</td>
</tr>
<tr>
<td></td>
<td>maps the request to the same server for all the subsequent messages using the sticky key. For</td>
</tr>
<tr>
<td></td>
<td>example, set StickyAVP “Session-Id”.</td>
</tr>
<tr>
<td>StickyCreationCmdList</td>
<td>Required; specifies the command list to create the sticky entries. Specify the list of ‘</td>
</tr>
<tr>
<td></td>
<td>separated command code, AVP name, and its value to create the sticky sessions.</td>
</tr>
<tr>
<td>StickyDeletionCmdList</td>
<td>Required; specifies the command list to delete the sticky entries. Specify the list of ‘</td>
</tr>
<tr>
<td></td>
<td>separated command code, AVP name, and its value to delete the sticky sessions.</td>
</tr>
</tbody>
</table>

Table 3-15 Diameter Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StickySessionKey</td>
<td>Required; used as the sticky key for mapping the sticky sessions. Set the value to a valid AVP</td>
</tr>
<tr>
<td></td>
<td>in order to use the sticky key for Diameter sessions. This ensures that Prime Access Registrar</td>
</tr>
<tr>
<td></td>
<td>maps the request to the same server for all the subsequent messages using the sticky key. For</td>
</tr>
<tr>
<td></td>
<td>example, set StickyAVP “Session-Id”.</td>
</tr>
<tr>
<td>StickyCreationCmdList</td>
<td>Required; specifies the command list to create the sticky entries. Specify the list of ‘</td>
</tr>
<tr>
<td></td>
<td>separated command code, AVP name, and its value to create the sticky sessions.</td>
</tr>
<tr>
<td>StickyDeletionCmdList</td>
<td>Required; specifies the command list to delete the sticky entries. Specify the list of ‘</td>
</tr>
<tr>
<td></td>
<td>separated command code, AVP name, and its value to delete the sticky sessions.</td>
</tr>
</tbody>
</table>
Table 3-15  Diameter Service Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>Required; default value is 0. Specifies the weight percentage for which the service needs to load balance the peer.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> When you set the weight to a value other than 0, the weight should be in multiples of 10 and the sum of the weights configured in the peer list should be equal to 100.</td>
</tr>
<tr>
<td>IMSIRanges</td>
<td>Required; used for load balancing. The value is set to comma separated values of IMSI Ranges.</td>
</tr>
<tr>
<td></td>
<td>For example, set IMSIRanges “112156000000001-112156001000000,112156010000001-112156011000000”</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> Prime Access Registrar uses the AVP configured in StickyAVP property to check whether the IMSI is in valid range.</td>
</tr>
<tr>
<td>IsActive</td>
<td>Optional; if this is set to true, the new sessions will not go to the peer server. By default, this is set as false.</td>
</tr>
</tbody>
</table>

You can use the Diameter Services List page for the following:

- Filtering Records
- Adding Diameter Service Details
- Editing Records
- Deleting Records

Adding Diameter Service Details

To add a new Diameter Service details:

**Step 1** Choose **Configuration > Services > Diameter**. The Diameter Services page is displayed.

**Step 2** Click **Add** to add new Diameter service details. The DIAMETER Services Details page is displayed.

**Step 3** Specify the required details in the **PEER Statements, Applications**, and **PEER Definitions Proxy** specific sections.

**Step 4** Click **Save DIAMETER Service** to save the specified details in the Diameter Services page. Otherwise click **Cancel** to return to the Diameter Services page without saving the details.

On successful creation of the Diameter Service properties, the Diameter Services page is displayed else a respective error message is displayed.

**Note** You may need to enter **PEER Statements, Applications**, and **PEER Definitions Proxy** details based on the **Role** that you select in the DIAMETER-Services page.
Adding the PEER Statements Details
To add new PEER Statement details:

Step 1 Click Add to add new PEER Statements details section. The fields specific to PEER Statements are displayed.

Step 2 Specify the required details.

Step 3 Click Save to save the specified details in the PEER Statements section. Otherwise click Cancel to return to the PEER Statements section without saving the details.

On successful creation of the Diameter Service properties, the Diameter Services page is displayed else a respective error message is displayed.

Adding the Applications Details
To add new Application details:

Step 1 Click Add to add new Applications details in the Application List section. The fields specific to Applications are displayed.

Step 2 Specify the required details.

Step 3 Click Save Appln to save the specified details in the Application List section. Otherwise click Cancel Appln to return to the Application List section without saving the details.

Adding the PEER Definitions Proxy Details
To add PEER Definitions Proxy details:

Step 1 Click Add to add new Proxy PEER Statements in the PEER Definitions Proxy section. The fields specific to Proxy PEER Statements are displayed.

Step 2 Specify the required details.

Step 3 Click Save to save the specified details in the Proxy PEER Statements section. Otherwise click Cancel to return to the Proxy PEER Statements section without saving the details.

CommandSets
A command set consists of commands and the action to perform during TACACS+ command authorization.

Adding a Command Set
To add a new command set:
Chapter 3      Using the Graphical User Interface

Configuring Cisco Prime Access Registrar

Step 1  Choose **Configuration > Command Sets**. Prime Access Registrar lists all the command sets available in the system. You can edit or delete an existing command set.

Step 2  Click **Add** to add a new command set.

Step 3  Enter a name and description for the command set.

Step 4  Provide the Command Set parameters. **Table 3-16** lists the parameters in the Add Command section.

### Table 3-16  Command Set Parameters

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>Select <strong>Permit</strong> or <strong>Deny</strong> to indicate the action to be performed on the command during TACACS+ command authorization.</td>
</tr>
<tr>
<td>Command</td>
<td>The command to add in the set. Example: <code>show</code></td>
</tr>
<tr>
<td>Arguments</td>
<td>The arguments for the command. Example: <code>~/serial*</code></td>
</tr>
</tbody>
</table>

**Note** Prime Access Registrar supports POSIX Extended Regular Expression (ERE) for command arguments.

Step 5  Click **Add** to add the new command to the set. The command details are displayed in the **Commands** section. You can edit or delete a command from the list as required.

Step 6  Click **Submit** to save the command set details.

You can use the Command Sets page to perform the following as well:

- Filtering Records
- Editing Records
- Deleting Records

**DeviceAccessRules**

A device access rule consists of conditions or expressions and the applicable command sets for TACACS+ command authorization.

**Adding a Device Access Rule**

To add a new device access rule:

Step 1  Choose **Configuration > Device Access Rules**. Prime Access Registrar lists all the device access rules available in the system. You can edit or delete an existing device access rule.

Step 2  Click **Add** to add a new device access rule.

Step 3  Enter a name and description for the device access rule.

Step 4  Choose the default device access action to perform on all commands in the device access rule. Options are **Permit All** or **Deny All**.
Step 5  In the Conditions field, include the expressions with **AND** or **OR** conditional operator.

Step 6  Select a command set from the drop-down list box and click **Add**. The selected command set is displayed in the Command Set Names list box available. Click **Delete** to remove any command set from the list.

Step 7  Provide the expression details for the device access rule. Table 3-17 lists the parameters for adding expressions.

<table>
<thead>
<tr>
<th>Field</th>
<th>Field Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the expression to include in the device access rule.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the expression.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Parameter to apply the condition on.</td>
</tr>
<tr>
<td>Value</td>
<td>Value of the parameter.</td>
</tr>
<tr>
<td>Note</td>
<td>Prime Access Registrar supports POSIX Extended Regular Expression (ERE) for condition expression value property.</td>
</tr>
</tbody>
</table>

Step 8  Click **Add** to add the expression to the list-box available in the Condition Expressions section. You can edit or delete the expression from the list as required.

Step 9  Click **Submit** to save the device access rule details.

### FastRules

FastRules provides a mechanism to easily choose the right authentication, authorization, accounting, and query service(s), drop, reject, or break flows, run a script, choose a session manager and/or a chain of fast rules required for processing a packet.

FastRules has the following capabilities:

- Provides maximum flexibility and ease in matching information in the incoming packets for choosing the appropriate service to apply
- Provides an option to match values in AVPs based on value ranges, exact match, and simple string comparisons using regex
- Provides easy and efficient alternative to rule/policy engine and scripting points for most common use cases—reduces the use of external scripts to choose an appropriate service

For more information about FastRules and the workflow, see Chapter 19, “Using FastRules to Process Packet Flow.”

### Adding a Fast Rule

To add a new fast rule:

Step 1  Choose **Configuration** > **FastRules**. Prime Access Registrar lists fast rules available for RADIUS, Diameter, and TACACS in the respective tabs. You can edit or delete an existing fast rule.

Step 2  Click **Add** to add a new fast rule. Table 3-18 provides the list of parameters in the FastRules Details page.
### Replication

The replication feature of Prime Access Registrar allows you to maintain identical configurations on multiple machines simultaneously. It eliminates the need to have administrators with multiple Prime Access Registrar installations, make the same configuration changes at each of their installations. Instead, only the master's configuration must be changed and the slave is automatically configured eliminating the need to make repetitive, error-prone configuration changes for each individual installation. In addition to enhancing server configuration management, using replication eliminates the need for a hot-standby machine.

Employing Prime Access Registrar's replication feature, both servers can perform RADIUS request processing simultaneously, eliminating wasted resources. It focuses on configuration maintenance only, not session information or installation-specific information.

**Table 3-19** lists and describes the fields in the Replication Details page.
Table 3-19  Replication Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General Properties tab</strong></td>
<td></td>
</tr>
<tr>
<td>Replication Type</td>
<td>Indicates the type of replication</td>
</tr>
<tr>
<td>Transaction Sync Interval (in ms)</td>
<td>Duration between periodic transmission of the TransactionSync message expressed in milliseconds. The default is 60000 or 1 minute.</td>
</tr>
<tr>
<td>Transaction Archive Limit</td>
<td>The default setting is 100. The value set for RepTransactionArchiveLimit should be the same on the master and the slave.</td>
</tr>
<tr>
<td>Replication Secret</td>
<td>The value of this setting must be identical on both the master and the slave.</td>
</tr>
<tr>
<td>Is Master</td>
<td>On the master, set RepIsMaster to TRUE. On the slave, set it to FALSE.</td>
</tr>
<tr>
<td>Master IP Address</td>
<td>Specifies the IP Address of the master.</td>
</tr>
<tr>
<td>Master Port</td>
<td>Specifies the port to be used to send replication messages to the master.</td>
</tr>
<tr>
<td>Replication IP Address</td>
<td>The value is set to the IP Address of the machine containing the Prime Access Registrar installation.</td>
</tr>
<tr>
<td>Replication Port</td>
<td>Defaults to port1645.</td>
</tr>
<tr>
<td><strong>Replication Members tab</strong></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Name of the slave. The name must be unique.</td>
</tr>
<tr>
<td>IP Address</td>
<td>Indicates the IP Address of the slave.</td>
</tr>
<tr>
<td>Port</td>
<td>Port upon which the master will send replication messages to the slave.</td>
</tr>
</tbody>
</table>

You can use the Replication Details page for the following:

- Filtering Records
- Adding Replication Details
- Adding the Replication Member Details
- Editing Records
- Deleting Records

**Adding Replication Details**

To add new replication details:

**Step 1** Choose Configuration > Replication. The Replication Details page is displayed.

**Step 2** Specify the replication details.

**Step 3** Enter the Replication Member Details, if needed.

**Step 4** Click Save to save the new replication details. Otherwise click Reset to restore the default values.
On successful creation of the replication details, a success message is displayed else a respective error message is displayed.

### Adding the Replication Member Details

To add new replication member details:

**Step 1** Click the **Replication Members** tab. The List of Replication Members section is displayed.

**Step 2** Enter the required details.

**Step 3** Click **Submit** to save the new replication member details.

---

### RADIUS Dictionary

The RADIUS dictionary passes information between a script and the RADIUS server, or between scripts running on a single packet.

Table 3-20 lists and describes the fields in the Add Radius Attributes page. The fields listed below are the entire list of all the available types. The fields are displayed based on the type selected.

**Table 3-20  RADIUS Dictionary Properties**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the RADIUS dictionary list</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the attribute</td>
</tr>
<tr>
<td>Attribute</td>
<td>Required; must be a number between 1-255. It must be unique within the Attribute dictionary list.</td>
</tr>
<tr>
<td>Type</td>
<td>Required; type governs how the value is interpreted and printed.</td>
</tr>
<tr>
<td>Minimum</td>
<td>Set to zero</td>
</tr>
<tr>
<td>Maximum</td>
<td>Set to 253</td>
</tr>
<tr>
<td>Enum Number</td>
<td>Enums allow you to specify the mapping between the value and the strings. After you have established this mapping, Prime Access Registrar then replaces the number with the appropriate string. The min/max properties represent the lowest to highest values of the enumeration.</td>
</tr>
<tr>
<td>Enum Equivalent</td>
<td>The value can range from 1 through 255. Click the Add button to save the details and list it in the Enums list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
<tr>
<td>Tag</td>
<td>The tag number value can range from 0 through 31. The default value is zero.</td>
</tr>
</tbody>
</table>
You can use the Radius Attributes page for the following:

- Filtering Records
- Adding RADIUS Dictionary Details
- Editing Records
- Deleting Records

**Adding RADIUS Dictionary Details**

To add new RADIUS dictionary details:

1. Choose **Configuration > Radius Dictionary**. The Radius Attributes page is displayed.
2. Click **Add** to add new RADIUS dictionary details. The Add RADIUS Dictionary page is displayed.
3. Enter the required details.
4. Click **Submit** to save the specified details in the Radius Attributes page. Otherwise click **Cancel** to return to the Radius Attributes page without saving the details.

On successful creation of the Radius Attributes, the Radius Attributes page is displayed else a respective error message is displayed.

**Vendor Dictionary**

The vendor dictionary allows the user to maintain the attributes of the vendor with respect to vendor id, vendor type and the attributes required to support the major NAS.

*Table 3-21* lists and describes the fields in the Add Vendor Dictionary page. The fields listed below are the entire list of all the available types. The fields are displayed based on the type selected.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Vendor dictionary list</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the attribute</td>
</tr>
<tr>
<td>Vendor ID</td>
<td>Required; must be a valid number and unique within the entire attribute dictionary</td>
</tr>
<tr>
<td>Type</td>
<td>Required; type governs how the value is interpreted and printed.</td>
</tr>
<tr>
<td>Minimum</td>
<td>Optional; set to zero</td>
</tr>
<tr>
<td>Maximum</td>
<td>Optional; set to 253</td>
</tr>
<tr>
<td>Enum Number</td>
<td>Optional; enums allow you to specify the mapping between the value and the strings. After you have established this mapping, Prime Access Registrar then replaces the number with the appropriate string. The min/max properties represent the lowest to highest values of the enumeration.</td>
</tr>
</tbody>
</table>
You can use the Vendor Dictionary page for the following:

- Filtering Records
- Adding Vendor Dictionary Details
- Editing Records
- Deleting Records

### Adding Vendor Dictionary Details

To add new vendor dictionary details:

1. Choose **Configuration > Vendor Dictionary**. The Vendor Attributes page is displayed.
2. Click **Add** to add new Vendor dictionary details. The Add Vendor Dictionary page is displayed.
3. Enter the required details.
4. Click **Submit** to save the specified details in the Vendor Attributes page. Otherwise click **Cancel** to return to the Vendor Attributes page without saving the details.

On successful creation of the vendor dictionary details, the Vendor Attributes page is displayed else a respective error message is displayed.

**Note**

After adding new vendor dictionary details, you can add vendor attributes details. Or you can also add vendor attributes details by clicking the link in the vendor dictionary list, see **Adding Vendor Attributes** for details.
Vendor Attributes

Vendor-specific attributes are included in specific RADIUS packets to communicate prepaid user balance information from the Prime Access Registrar server to the AAA client, and actual usage, either interim or total, between the NAS and the Prime Access Registrar server.

Table 3-22 lists and describes the fields in the Add Vendor Attributes page.

### Table 3-22 Vendor Attribute Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Vendor attribute list</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the attribute</td>
</tr>
<tr>
<td>Attribute</td>
<td>Required; must be a valid number and unique within the entire attribute dictionary</td>
</tr>
<tr>
<td>Type</td>
<td>Required; type governs how the value is interpreted and printed.</td>
</tr>
<tr>
<td>Minimum</td>
<td>Optional; set to zero</td>
</tr>
<tr>
<td>Maximum</td>
<td>Optional; set to 253</td>
</tr>
<tr>
<td>Enum Number</td>
<td>Optional; enums allow you to specify the mapping between the value and the strings. After you have established this mapping, Prime Access Registrar then replaces the number with the appropriate string. The min/max properties represent the lowest to highest values of the enumeration.</td>
</tr>
<tr>
<td>Enum Equivalent</td>
<td>Optional; the value can range from 1 through 255. Click the Add button to save the details and list it in the Enums list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
<tr>
<td>Tag</td>
<td>Optional; the tag number value can range from 0 through 31. The default value is zero.</td>
</tr>
</tbody>
</table>

You can use the Vendor Attributes page for the following:

- Filtering Records
- Adding Vendor Attributes
- Editing Records
- Deleting Records

### Adding Vendor Attributes

To add new Vendor attributes:

**Step 1** Choose **Configuration > Vendor Dictionary**. The Vendor Attributes page is displayed.

**Step 2** Click the Vendor name link. The Vendor Attributes page is displayed.

**Step 3** Click **Add** to add new Vendor attributes. The Add Vendor Attributes page is displayed.
Step 4  Enter the required details.

Step 5  Click **Submit** to save the specified details in the Vendor Attributes page. Otherwise click **Cancel** to return to the Vendor Attributes page without saving the details.

On successful creation of the vendor attributes, the Vendor Attributes page is displayed else a respective error message is displayed.

---

**Vendors**

The **Vendor** object provides a central location for specifying all of the request and response processing a particular NAS or Proxy vendor requires. Depending on the vendor, it might be necessary to map attributes in the request from one set to another, or to filter out certain attributes before sending the response to the client. For more information about standard RADIUS attributes, see Chapter C, “RADIUS Attributes.”

**Note** When you have also set /Radius/IncomingScript, Cisco Prime Access Registrar runs that script before the vendor’s script. Conversely, when you have set a /Radius/Outgoing script, Cisco Prime Access Registrar runs the vendor’s script before that script.

**Table 3-23** lists and describes the fields in the Add Vendor page.

<table>
<thead>
<tr>
<th><strong>Vendor Properties</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fields</strong></td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>IncomingScript</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>OutgoingScript</td>
</tr>
</tbody>
</table>

You can use the Vendors page for the following:

- Filtering Records
- Adding Vendor Details
- Editing Records
- Deleting Records

**Adding Vendor Details**

To add new Vendor details:
Step 1 Choose Configuration > Vendors. The Vendors page is displayed.

Step 2 Click Add to add new Vendor details. The Add Vendor page is displayed.

Step 3 Enter the required details.

Step 4 Click Submit to save the specified details in the Vendors page. Otherwise click Cancel to return to the Vendors page without saving the details.

On successful creation of the vendor details, the Vendors page is displayed else a respective error message is displayed.

Translating

Translations add new attributes to a packet or change an existing attribute from one value to another. The Translations subdirectory lists all definitions of Translations the RADIUS server can apply to certain packets.

Under the /Radius/Translations directory, any translation to insert, substitute, or translate attributes can be added. The following is a sample configuration under the /Radius/Translations directory:

```
cd /Radius/Translations
Add T1
cd T1
Set DeleAttrs Session-Timeout,Called-Station-Id
cd Attributes
Set Calling-Station-Id 18009998888
```

DeleAttrs is the set of attributes to be deleted from the packet. Each attribute is comma separated and no spaces are allowed between attributes. All attribute value pairs under the attributes subdirectory are the attributes and values that are going to be added or translated to the packet.

Under the /Radius/Translations/T1/Attributes directory, inserted or translated attribute value pairs can be set. These attribute value pairs are either added to the packet or replaced with the new value.

If a translation applies to an Access-Request packet, by referencing the definition of that translation, the Prime Access Registrar server modifies the Request dictionary and inserts, filters, and substitutes the attributes accordingly. You can set many translations for one packet and the Prime Access Registrar server applies these translations sequentially.

**Note**

Later translations can overwrite previous translations.

**Table 3-24** lists and describes the fields in the Add Translations page.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td>Required; must be unique in the Translations list.</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Optional; description of the Translation</td>
</tr>
</tbody>
</table>
### Adding Translation Details

To add new translation details:

1. **Step 1** Choose **Configuration > Translations**. The Translations page is displayed.
2. **Step 2** Click **Add** to add new translations details. The Add Translations page is displayed.
3. **Step 3** Enter the required details.
4. **Step 4** Click **Add Translation** to save the specified details in the Translations page. Otherwise click **Cancel** to return to the Translations page without saving the details.

On successful creation of the translation details, the Translations page is displayed else a respective error message is displayed.

---

**Table 3-24 Translations Properties (continued)**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Type</td>
<td>Optional; select either <strong>RADIUS</strong> or <strong>VENDOR</strong>. If Vendor is selected, specify the vendor type from the drop-down list. Select the attributes from the available list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See <strong>Relocating Records</strong> for more details.</td>
</tr>
</tbody>
</table>

**Attributes tab**

<table>
<thead>
<tr>
<th>Attribute Type</th>
<th>Optional; select either <strong>RADIUS</strong> or <strong>VENDOR</strong>. If Vendor is selected, specify the vendor type from the drop-down list.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Name</td>
<td>Optional; based on the Attribute Type selected, the attribute name is automated. Set the relevant name for the attribute type selected.</td>
</tr>
<tr>
<td>Attribute Value</td>
<td>Optional; set the value for the selected attribute. Click the <strong>Add</strong> button to save the details and list it in Radius and Value list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See <strong>Relocating Records</strong> for more details. To delete the available attributes, select the relevant attribute and click the <strong>Delete</strong> button below.</td>
</tr>
</tbody>
</table>

You can use the Translations page for the following:

- Filtering Records
- Adding Translation Details
- Editing Records
- Deleting Records
TranslationGroups

You can add translation groups for different user groups under TranslationGroups. All Translations under the Translations subdirectory are applied to those packets that fall into the groups. The groups are integrated with the Prime Access Registrar Rule engine.

The Prime Access Registrar Administrator can use any RADIUS attribute to determine the Translation Group. The incoming and outgoing translation group can be different translation groups. For example, you can set one translation group for incoming translations and one for outgoing translations.

Under the /Radius/TranslationGroups directory, translations can be grouped and applied to certain sets of packets, which are referred to in a rule. The following is a sample configuration under the /Radius/TranslationGroups directory:

```
cd /Radius/TranslationGroups
Add CiscoIncoming
cd CiscoIncoming
cd Translations
Set 1 T1
```

The translation group is referenced through the Prime Access Registrar Policy Engine in the /Radius/Rules/<RuleName>/Attributes directory. Incoming-Translation-Groups are set to a translation group (for example CiscoIncoming) and Outgoing-Translation-Groups to another translation group (for example CiscoOutgoing).

Table 3-25 lists and describes the fields in the Add Translation Groups page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Translations list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the Translation Group.</td>
</tr>
<tr>
<td>Translations</td>
<td>Optional; lists of translation. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details.</td>
</tr>
</tbody>
</table>

You can use the Translation Groups page for the following:

- Filtering Records
- Adding Translation Group Details
- Editing Records
- Deleting Records

Adding Translation Group Details

To add new translation group details:

**Step 1** Choose Configuration > TranslationGroups. The Translation Groups page is displayed.

**Step 2** Click Add to add new translation group details. The Add TranslationGroup page is displayed.

**Step 3** Enter the required details.
Step 4  Click **Add TranslationGroup** to save the specified details in the Translation Groups page. Otherwise click **Cancel** to return to the Translation Groups page without saving the details.

On successful creation of the translation group details, the Translation Groups page is displayed else a respective error message is displayed.

---

**Diameter**

Diameter is a computer networking protocol for Authentication, Authorization and Accounting (AAA). It is a successor to RADIUS or an enhanced version of the RADIUS protocol. It includes numerous enhancements in all aspects, such as error handling and message delivery reliability. It extracts the essence of the AAA protocol from RADIUS and defines a set of messages that are general enough to be the core of the Diameter Base protocol. The various applications that require AAA functions can define their own extensions on top of the Diameter base protocol, and can benefit from the general capabilities provided by the Diameter base protocol.

The following sections can be used to configure Diameter transport management properties, session management properties, add new application, commands associated with it and application specific AVPs:

- General
- Session Management
- Applications
- Commands
- DiameterAttributes

**General**

This section explains how to set Diameter general configuration such as product name, version, and transport management properties.

**Setting General Diameter Parameters**

Table 3-26 lists and describes the fields in the General Diameter Properties page.

**Table 3-26  General Diameter Properties**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General section</strong></td>
<td></td>
</tr>
<tr>
<td>Product</td>
<td>Optional; name of the product.</td>
</tr>
<tr>
<td>AuthApplicationIdList</td>
<td>Specifies the list of AuthApplications that the Prime Access Registrar server registers to Diameter Base stack during start up. It is a combination of Auth ApplicationId’s separated by a colon.</td>
</tr>
<tr>
<td>Version</td>
<td>Optional; version number.</td>
</tr>
</tbody>
</table>
Table 3-26  General Diameter Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcctApplicationIdList</td>
<td>Specifies the list of AcctApplications that the Prime Access Registrar server registers to Diameter Base stack during start up. It is a combination of Acct ApplicationId's separated by a colon.</td>
</tr>
</tbody>
</table>

**Transport Management section**

<table>
<thead>
<tr>
<th>Identity</th>
<th>Required; identity of the system on which Diameter application is running. Must be set to a valid resolvable string.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Realm</td>
<td>Required; must be set to a valid Realm in the domain.</td>
</tr>
<tr>
<td>EnableIPV6</td>
<td>Required; if set to TRUE it enables IPV6 for the Diameter application.</td>
</tr>
<tr>
<td>ValidateIncomingMessages</td>
<td>Check the box to validate incoming messages.</td>
</tr>
<tr>
<td>ValidateOutgoingMessages</td>
<td>Check the box to validate outgoing messages.</td>
</tr>
<tr>
<td>MaximumNumberofDiameterPackets</td>
<td>Required; the maximum number of Diameter packets that can be processed.</td>
</tr>
<tr>
<td>DiameterPacketSize</td>
<td>Required; the Diameter packet size that can be processed. An incoming Diameter packet with a packet size more than the value set in this field will be dropped.</td>
</tr>
<tr>
<td>WatchdogTimeout</td>
<td>Required; specifies the time interval between watch dog messages.</td>
</tr>
<tr>
<td>ReserveDiameterPacketPool</td>
<td>Percentage of the Diameter packet pool to reserve for the Diameter remote server responses.</td>
</tr>
<tr>
<td>TCPListenPort</td>
<td>Required; port number on which the Prime Access Registrar server listens for TCP peer connections.</td>
</tr>
<tr>
<td>SCTPListenPort</td>
<td>Required; port number on which the Prime Access Registrar server listens for SCTP peer connections.</td>
</tr>
<tr>
<td>ReconnectInterval</td>
<td>Required; specifies the time interval between which Prime Access Registrar server attempts to connect to a disconnected peer. If set to 0, then no attempt will be made to connect to a disconnected peer.</td>
</tr>
<tr>
<td>MaxReconnections</td>
<td>Required; specifies the number of times Prime Access Registrar server tries to make a reconnection attempt. If set to 0, then no attempt will be made to reconnect.</td>
</tr>
<tr>
<td>RequestRetransmissionInterval</td>
<td>Required; the time for which retransmission of pending requests will be done. If set to 0, then no attempt will be made to retransmit.</td>
</tr>
<tr>
<td>MaxRequestRetransmissionCount</td>
<td>Required, maximum number of times Prime Access Registrar server tries to retransmit a pending request. If set to 0, then no attempt will be made to retransmit.</td>
</tr>
</tbody>
</table>
Setting Up the General Diameter Parameters

To set up the general Diameter parameters:

**Step 1** Choose **Configuration > Diameter > General**. The General Diameter page is displayed.

**Step 2** Specify the required details.

**Step 3** Click **Set** to save the specified details.

On successful creation of the general Diameter parameters, a success message is displayed else a respective error message is displayed.

**Session Management**

Diameter Base protocol stack provides the functionality of Session Management. Base Stack maintains sessions separately for authentication and accounting messages. Session-Id AVP is used to identify the user session.

Table 3-27 lists and describes the fields in the Session Management page.

---

**Table 3-26  General Diameter Properties (continued)**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receive BufferSize</td>
<td>Required; initial size of buffer that is preallocated for message reception.</td>
</tr>
</tbody>
</table>

**SCTPOptions Section**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxInitRetry</td>
<td>Maximum number of retries for INIT message to open a connection. Valid range is 0 - 255. Set to 0 to retry indefinitely.</td>
</tr>
<tr>
<td>MaxInboundStream</td>
<td>Maximum number of incoming streams per connection. Valid range is 1 - 65545.</td>
</tr>
<tr>
<td>MaxOutboundstream</td>
<td>Maximum number of outgoing streams per connection. Valid range is 1 - 65545.</td>
</tr>
<tr>
<td>HeartbeatInterval</td>
<td>Default heartbeat interval for a connection.</td>
</tr>
<tr>
<td>EnableHeartbeat</td>
<td>Indicates whether to enable or disable heartbeat to monitor the connections and allow earlier detection of loss connections.</td>
</tr>
<tr>
<td>AdvertisedHostName</td>
<td>Optional, specifies the local hostname address that will be advertised by the Prime Access Registrar server to other peers during CER/CEA exchange. For example: AdvertisedHostNames = toby-ar1.cisco.com</td>
</tr>
</tbody>
</table>
To set up the session management properties:

**Step 1**  Choose **Configuration > Diameter>SessionManagement.** The Session Management page is displayed.

**Step 2**  Enter the required details and click **Set.**

On successful creation of the parameters, a success message is displayed else a respective error message is displayed.
Applications

A Diameter application is not a software application, but a protocol based on the Diameter base protocol (defined in RFC 6733). Each application is defined by an application identifier and can add new command codes and/or new mandatory AVPs.

When you click the Add button in the Applications page, the Application Details page is displayed. Table 3-28 lists and describes the fields in the Application Details page.

<table>
<thead>
<tr>
<th>Table 3-28</th>
<th>Diameter Application Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fields</td>
<td>Description</td>
</tr>
<tr>
<td>Name</td>
<td>Required; name of the application.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the application.</td>
</tr>
<tr>
<td>VendorSpecific</td>
<td>Required; the default is FALSE. If set to FALSE, the application is ordinary application and user is prompted to enter the ApplicationID. If set to TRUE, the application is a VendorSpecific Application. User is prompted to enter VendorSpecificApplicationID and VendorID.</td>
</tr>
<tr>
<td>AuthApplication</td>
<td>Required; if set to TRUE the application represents AuthApplication else it represents Accounting Application.</td>
</tr>
<tr>
<td>ApplicationID</td>
<td>Required; specifies the unique integer value for the application. The following are examples of Diameter application: NASREQ 1 Mobile-IP 2 Diameter Base Accounting 3</td>
</tr>
<tr>
<td>Note</td>
<td>ApplicationURI property must be set to 0 for Base Protocol.</td>
</tr>
<tr>
<td>VendorSpecificApplicationID</td>
<td>Required; specifies the integer value for the vendor specific application.</td>
</tr>
<tr>
<td>VendorID</td>
<td>Required; specifies the VendorID for the application. Example: DIAMETER 3GPP Cx APPLICATION VendorSpecificApplicationID 16777216 VendorID 10415</td>
</tr>
<tr>
<td>Commands</td>
<td>Required; an indexed list from 1 to &lt;n&gt;. Each entry in the list is the name of the command. It specifies the list of commands associated with the application. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details.</td>
</tr>
</tbody>
</table>

You can use the Applications page for the following:
- Filtering Records
• Adding Diameter Application Details
• Commands
• Editing Records
• Deleting Records

Adding Diameter Application Details

To add new Diameter application details:

**Step 1** Choose **Configuration > Diameter > Applications**. The Applications page is displayed.

**Step 2** Click **Add**. The Application Details page is displayed.

**Step 3** Enter the relevant details.

**Step 4** Click **Add Application** to save the specified details in the Application Details page. Otherwise click **Cancel** to return to the Applications page without saving the details.

On successful creation of the Applications details, a success message is displayed else a respective error message is displayed.

Commands

Each command in Diameter is associated with a command code. The command can be a request command or an answer command which is identified by the ‘R’ bit in the Command Flags field of the Diameter header.

When you click the Add button in the commands page, the Command Details page is displayed. **Table 3-29** lists and describes the fields in the Command Details page.

<table>
<thead>
<tr>
<th>Table 3-29 Diameter Commands Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fields</strong></td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Command Code</td>
</tr>
<tr>
<td>EnableProxyBit</td>
</tr>
<tr>
<td>RequestFixed tab</td>
</tr>
<tr>
<td>RequestRequired tab</td>
</tr>
<tr>
<td>RequestOptional tab</td>
</tr>
<tr>
<td>AnswerFixed tab</td>
</tr>
</tbody>
</table>
Configuring Cisco Prime Access Registrar

You can click the Add button in the Command Details page to add the AVP details. Table 3-30 lists and describes the fields displayed on clicking the Add button.

### DiameterAttributes

You can define the attributes to use in the Diameter EAP application.

Table 3-31 lists and describes the fields in the DiameterAttributes page.
Adding Diameter Attributes

To add the Diameter attributes:

Step 1  Choose Configuration > Diameter > DiameterAttributes. The DiameterAttributes page is displayed.
Step 2  Click Add.
Step 3  Provide the relevant details as explained in Table 3-31.
Step 4  Click Add DiameterAttributes to save the specified details. Otherwise click Cancel to return to the previous page without saving the details.

The DiameterAttributes page is displayed with the newly added details or a respective error message is displayed.

Advanced

Advanced objects allow configuring system-level properties and the Attribute dictionary. Under normal system operation, the system-level properties should not be changed.

The following list helps you in defining the system-level properties and attribute dictionary:

- Default
- BackingStore/ServerParam
- RemoteSessionServer
- SNMP
- DDNS
**Default**

This feature of GUI allows you in configuring the default values for other functionalists of GUI. The configurations set in this feature reflects on all the other features.

Table 3-32 lists and describes the fields in the Default Advanced Details page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAAFileServiceSyncInterval</td>
<td>Required; specified in milliseconds, the default is 75. This property governs how often the file AAA service processes accounting requests and writes the accounting records to the file. You can lower the number to reduce the delay in acknowledging the Accounting-Requests at the expense of more frequent flushing of the accounting file to disk. You can raise the number to reduce the cost of flushing to disk, at the expense of increasing the delays in acknowledging the Accounting-Requests. The default value was determined to provide a reasonable compromise between the two alternatives.</td>
</tr>
<tr>
<td>RemoteRadiusServerInterface</td>
<td>When set, specifies the local interface to bind to when creating the RemoteRadiusServer socket. If not set, the Prime Access Registrar binds to IPADDR_ANY.</td>
</tr>
<tr>
<td>MaximumNumberOfXML-Packets</td>
<td>Required when using identity caching. Indicates the maximum number of XML packets to be sent or received. The minimum value is 1 and the maximum is a 32-bit unsigned integer. The default is 1024.</td>
</tr>
<tr>
<td>MaximumODBCResultSize</td>
<td>Required; specifies maximum size in bytes for an ODBC mapping. This parameter affects both ODBC result sizes and the trace log buffer for tracing script calls that access any of the dictionaries. (Default value is 256.)</td>
</tr>
<tr>
<td>XMLUDPPacketSize</td>
<td>Required when using identity caching. Indicates the maximum size of XML packets to be sent or received. The minimum value is 1 and the maximum is a 32-bit unsigned integer. The default is 4096.</td>
</tr>
<tr>
<td>InitialBackgroundTimerSleepTime</td>
<td>Required; the default is 5. This property specifies the amount of time the time queue should initially sleep before beginning processing. This property is only used for initial synchronization and should not be changed.</td>
</tr>
</tbody>
</table>
### Table 3-32 Default Configuration Details (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RemoteLDAPServerThreadTimerInterval</td>
<td>Required; specified in milliseconds, the default is 10. This property governs how often the Ipd RemoteServer thread checks to see if any results have arrived from the remote LDAP server. You can modify it to improve the throughput of the server when it proxies requests to a remote LDAP server.</td>
</tr>
<tr>
<td>AdvancedDuplicateDetectionMemoryInterval</td>
<td>Required when the Advanced Duplicate Detection feature is enabled. This property specifies how long (in milliseconds) Cisco Prime Access Registrar should remember a request. You must specify a number greater than zero. The default is 10,000.</td>
</tr>
<tr>
<td>RollingEncryptionKeyChangePeriod</td>
<td>Used in conjunction with the session-cache Resource Manager, this property specifies the length of time a given EncryptionKey will be used before a new one is created. When the session-cache Resource Manager caches User-Password attributes, Prime Access Registrar encrypts the User-Password so it is not stored in memory or persisted on disk in clear text. Prime Access Registrar uses up to 255 encryption keys, using a new one after each RollingEncryptionKeyChangePeriod expires. If RollingEncryptionKeyChangePeriod is set to 2 days, Prime Access Registrar will create and begin using a new EncryptionKey every two days. The oldest key will be retired, and Prime Access Registrar will re-encrypt any User-Passwords that used the old key with the new key. This way, if the RollingEncryptionKeyChangePeriod is set to 1 day, no key will be older than 255 days.</td>
</tr>
<tr>
<td>DefaultReturnedSubnetSizeIfNoMatch</td>
<td>Optional; used with the ODAP feature and reflects the returned size of the subnet if no matched subnet is found. There are three options to select if an exactly matched subnet does not exist: Bigger, Smaller, and Exact. The default is Bigger.</td>
</tr>
<tr>
<td>ODBCEnvironmentMultiValueDelimiter</td>
<td>Optional; allows you to specify a character that separates multivalued attributes in the marker list when using Oracle (or ODBC) accounting.</td>
</tr>
<tr>
<td>RemoteSigtranServerThreadTimerInterval</td>
<td>Required; specified in milliseconds, the default is 10. This property governs how often the sigtran RemoteServer thread checks to see if any results have arrived from the remote HLR/AuC server. You can modify it to improve the throughput of the server when it proxies requests to a remote sigtran server.</td>
</tr>
<tr>
<td>AdditionalNativeOracleErrors</td>
<td>Optional; 5 digit Oracle native error in order to disconnect the ODBC/OCI remote servers.</td>
</tr>
<tr>
<td>EnableLengthFlag</td>
<td>Check this box to enable the length flag.</td>
</tr>
<tr>
<td>FlushDiskInBackground</td>
<td>Check this box to allow Prime Access Registrar to flush the accounting record to disk before it acknowledges the request packets.</td>
</tr>
<tr>
<td>SendOpCodeInISDResponse</td>
<td>Check this box to send operator code in the ISD response.</td>
</tr>
<tr>
<td>EnableRoutingContextInM3UA</td>
<td>Check this box to enable routing context in M3UA.</td>
</tr>
<tr>
<td>DefaultRadiusSharedSecret</td>
<td>Enter the default shared secret for RADIUS server.</td>
</tr>
<tr>
<td>ReserveRADIUSPacketPool</td>
<td>Percentage of the RADIUS packet pool to reserve for the RADIUS remote server responses.</td>
</tr>
</tbody>
</table>
### Table 3-32 Default Configuration Details (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TLSv1Enabled</td>
<td>Applicable only for Diameter; Set to TRUE to use TLS version 1.0 and above for Diameter connection. Set to FALSE to use TLS version greater than 1.0 for Diameter connection.</td>
</tr>
</tbody>
</table>
### AR Flags section

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HideSharedSecretAndPrivateKeys</td>
<td>Optional; the default value is TRUE. The HideSharedSecretAndPrivateKeys property hides:</td>
</tr>
<tr>
<td></td>
<td>• The secret that is shared between a RADIUS Client and a RADIUS Server or between two RADIUS servers in a RADIUS proxy scenario.</td>
</tr>
<tr>
<td></td>
<td>• The PrivateKeyPassword under the certificate-based EAP services.</td>
</tr>
<tr>
<td></td>
<td>When this property is set to TRUE, the following properties are displayed as <code>&lt;encrypted&gt;</code>:</td>
</tr>
<tr>
<td></td>
<td>• PrivateKeyPasswords in:</td>
</tr>
<tr>
<td></td>
<td>– peap-v0 service</td>
</tr>
<tr>
<td></td>
<td>– peap-v1 service</td>
</tr>
<tr>
<td></td>
<td>– eap-tls service</td>
</tr>
<tr>
<td></td>
<td>– eap-ttls service</td>
</tr>
<tr>
<td></td>
<td>– eap-fast service</td>
</tr>
<tr>
<td></td>
<td>• SharedSecret in:</td>
</tr>
<tr>
<td></td>
<td>– RemoteServers of type RADIUS</td>
</tr>
<tr>
<td></td>
<td>– RemoteServers of type map-gateway</td>
</tr>
<tr>
<td></td>
<td>– Clients object</td>
</tr>
<tr>
<td></td>
<td>– Resource Manager of type usr-vpn under Gateway subobject</td>
</tr>
<tr>
<td></td>
<td>• PseudonymSecret in eap-sim service</td>
</tr>
<tr>
<td></td>
<td>• DynamicAuthSecret under DynamicAuthorizationServer subject in Clients object</td>
</tr>
<tr>
<td></td>
<td>• RepSecret under Replication</td>
</tr>
<tr>
<td></td>
<td>• Secret in <code>/radius/advanced/DDNS/TSIGKeys</code></td>
</tr>
<tr>
<td></td>
<td>When the value for this property is set to FALSE, all the above properties are displayed in clear text.</td>
</tr>
<tr>
<td>ListenForDynamicAuthorizationRequests</td>
<td>Must be set to TRUE when using the Change of Authorization (CoA) feature or Packet of Disconnect (POD) feature. Default is FALSE.</td>
</tr>
<tr>
<td>RequireNASsBehindProxy-BeInClientList</td>
<td>Optional; the default is FALSE. If you accept the default, Cisco Prime Access Registrar only uses the source IP address to identify the immediate client that sent the request. Leaving it FALSE is useful when this RADIUS Server should only know about the proxy server and should treat requests as if they came from the proxy server. This might be the case with some environments that buy bulk dial service from a third party and thus do not need to, or are unable to, list all of the NASs behind the third party’s proxy server. When you set it to TRUE, you must list all of the NASs behind the Proxy in the Clients list.</td>
</tr>
</tbody>
</table>
Use Advanced Duplicate Detection

- **Required:** the default is FALSE. Set this property to TRUE when you want Cisco Prime Access Registrar to use a more robust duplicate request filtering algorithm.

Detect Out Of Order Accounting Packets

- **Optional:** used to detect accounting packets that arrive out of sequential order. The default is FALSE. This property is useful when using accounting and session management in a RADIUS proxy service.

  When the Detect Out Of Order Accounting Packet property is enabled (set to TRUE), a new **Class** attribute is included in all outgoing Accept packets. The value for this Class attribute will contain the session magic number. The client will echo this value in the accounting packets, and this will be used for comparison.

  The session magic number is a unique number created for all sessions when the session is created or reused and the Detect Out Of Order Accounting Packet property is set to TRUE. The Detect Out Of Order Accounting Packet property is used to detect out-of-order Accounting-Stop packets in roaming scenarios by comparing the session magic number value in the session with the session magic number value contained in the Accounting packet.

  The value of 0xffffffff is considered by the Prime Access Registrar server to be a wild card magic number. If any accounting stop packets contain the value of 0xffffffff, it will pass the session magic validation even if the session’s magic number is something else.

  The format of the class attribute is as follows:

  `<4-byte Magic Prefix><4-byte server IP address><4-byte Magic value>`

Java and EAP Parameters section

Classpath For Java Extensions

- A string which is the classpath to be used to locate Java classes and jar files containing the classes required for loading the Java extensions, either Java extension points or services.

  **Note** The classpath will always contain the directory `{INSTALL-DIR}/scripts/radius/java` and all of the jar files in that directory.

Java VMO Options

- A string that can contain options to be passed to the JRE upon startup. Java VMO Options should be used only when requested by Cisco TAC.
**Table 3-32  Default Configuration Details (continued)**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EapBadMessagePolicy</td>
<td>Set to one of two values: SilentDiscard (the default) or RejectFailure.</td>
</tr>
<tr>
<td></td>
<td>When set to SilentDiscard, the Prime Access Registrar server silently</td>
</tr>
<tr>
<td></td>
<td>discards and ignores bad EAP messages unless the protocol specification</td>
</tr>
<tr>
<td></td>
<td>explicitly requires a failure message.</td>
</tr>
<tr>
<td></td>
<td>When set to RejectFailure, the Prime Access Registrar server sends</td>
</tr>
<tr>
<td></td>
<td>RADIUS Access-Rejects messages with embedded EAP-Failure in response to bad</td>
</tr>
<tr>
<td></td>
<td>EAP messages as described in Internet RFC 3579.</td>
</tr>
<tr>
<td>CertificateDBPath</td>
<td>Required if you are using an LDAP RemoteServer and you want Prime Access</td>
</tr>
<tr>
<td></td>
<td>Registrar to use SSL when communicating with that LDAP RemoteServer. This</td>
</tr>
<tr>
<td></td>
<td>property specifies the path to the directory containing the client certificates</td>
</tr>
<tr>
<td></td>
<td>to be used when establishing an SSL connection to an LDAP RemoteServer. This</td>
</tr>
<tr>
<td></td>
<td>directory must contain the cert7.db and cert5.db certificates and the key3.db</td>
</tr>
<tr>
<td></td>
<td>and key.db files database used by Netscape Navigator 3.x (and above) or the</td>
</tr>
<tr>
<td></td>
<td>ServerCert.db certificate database used by Netscape 2.x servers.</td>
</tr>
</tbody>
</table>

**Setting Default Configuration**

To set up the default configuration details:

**Step 1** Choose **Configuration > Advanced > Default**. The Default Advanced Details page is displayed.

**Step 2** Enter the relevant details.

**Step 3** Click **Set** to save the specified details in the Default Advanced Details page. Otherwise, click **Reset** to restore the default values. On successful creation of the default configurations, a success message is displayed else a respective error message is displayed.

**BackingStore/ServerParam**

The Backing Store is a Parsing Tool which helps you in analyzing the session backing store files. It retrieves the information on RADIUS sessions, clears phantom sessions details manually and processes the binary log files information to user-readable format.

The Server parameters are set to configure objects to remote server using the relevant aregcmd commands.
Table 3-33 lists and describes the fields in the Backing/ServerParam Advanced Details page.

**Table 3-33  BackingStore/ServerParameter Properties**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Backing Store section</strong></td>
<td></td>
</tr>
<tr>
<td>SessionBackingStoreSyncInterval</td>
<td>Sessions will be written to the backing store at this interval.</td>
</tr>
<tr>
<td>PacketBackingStoreSyncInterval</td>
<td>The minimum value is 1 and the maximum is a 32-bit unsigned integer. The default is 75.</td>
</tr>
<tr>
<td>SessionBackingStorePruneInterval</td>
<td>Required; specifies the sleep time interval of the session backing store pruning thread. The recommended and default value is 6 hours, but you can modify this based on the traffic patterns you experience.</td>
</tr>
<tr>
<td></td>
<td>With SessionBackingStorePruneInterval set to 6 hours, pruning will occur 6 hours after you restart or reload the Prime Access Registrar server and recur every 6 hours.</td>
</tr>
<tr>
<td></td>
<td>You can set a very low value for this property to make pruning continuous, but there might not be enough data accumulated for the pruning to occur and pruning might be less effective compared to the default setting.</td>
</tr>
<tr>
<td>PacketBackingStorePruneInterval</td>
<td>Required; specifies the sleep time interval of the packet backing store pruning thread. The recommended value is 6 hours, but you can modify this based on the traffic patterns you experience.</td>
</tr>
<tr>
<td></td>
<td>When PacketBackingStorePruneInterval is set to 6 hours, pruning will occur 6 hours after you restart or reload the Prime Access Registrar server and recur every 6 hours.</td>
</tr>
<tr>
<td></td>
<td>You can set a very low value for this property to make pruning continuous, but there might not be enough data accumulated for the pruning to occur and pruning might be less effective compared to the default setting.</td>
</tr>
<tr>
<td>BackingStoreDiscThreshold</td>
<td>Required; the default is 10 gigabytes. The value of BackingStoreDiscThreshold is made up of a number of units which can be K, kilobyte, or kilobytes, M, megabyte, or megabytes, or G, gigabyte, or gigabytes.</td>
</tr>
<tr>
<td></td>
<td>BackingStoreDiscThreshold is used with session management and ODBC accounting and ensures that any data log files generated will not cross the BackingStoreDiscThreshold.</td>
</tr>
</tbody>
</table>
Table 3-33  BackingStore/ServerParameter Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SessionPurgeInterval</td>
<td>Optional; the SessionPurgeInterval property determines the time interval at which to check for timed-out sessions. If no value is set, the session timeout feature is disabled. The checks are performed in the background when system resources are available, so checks might not always occur at the exact time set. The minimum recommended value for SessionPurgeInterval is 60 minutes. The SessionPurgeInterval value is comprised of a number and a units indicator, as in \text{n units}, where a unit is one of minutes, hours, days, or weeks.</td>
</tr>
<tr>
<td>StaleSessionTimeout</td>
<td>Required; the default value is “1 hour.” Specifies the time interval to maintain a session when a client does not respond to Accounting-Stop notification. When the Prime Access Registrar server does not receive an Accounting-Response from a client after sending an Accounting-Stop packet, Prime Access Registrar maintains the session for the time interval configured in this property before releasing the session. This property is stored as a string composed of two parts: a number and a unit indicator (&lt;\text{n}&gt; &lt;\text{units}&gt;) similar to the MaxFileAge property where the unit is one of: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, or Weeks.</td>
</tr>
<tr>
<td>NumberOfRadiusIdentifiersPerSocket</td>
<td>This represents the number of RADIUS Identifiers that Prime Access Registrar can use per source port, while proxying requests to remote servers. To use a different source port for every request that is proxied, you need to set the value of this property to one.</td>
</tr>
<tr>
<td>EnableStickySessionCount</td>
<td>Required; either True or False and the default value is True. When set to True, Prime Access Registrar displays the peer specific stats showing the number of sticky sessions associated with a peer for Diameter proxy service in name_radius_log file.</td>
</tr>
<tr>
<td>StickySessionCountInterval</td>
<td>Required; specified in milliseconds and the default is 60000. When the EnableStickySessionCount is set to True, this field specifies how often the Diameter proxy service will display the number of sticky sessions associated with a peer.</td>
</tr>
<tr>
<td>StickySessionSyncInterval</td>
<td>Required; specified in milliseconds and the default value is 500. Specifies how often the Diameter proxy service will write the sticky sessions to a file located in /cisco-ar/temp/__sticky_sessions_store location.</td>
</tr>
</tbody>
</table>
### Table 3-33  BackingStore/ServerParameter Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Server Parameters section</strong></td>
<td></td>
</tr>
<tr>
<td>MaximumNumberOfRadiusPackets</td>
<td>Required; the default is 8192. This is a critical property you should set high enough to allow for the maximum number of simultaneous requests. When more requests come in than there are packets allocated, Cisco Prime Access Registrar will drop those additional requests.</td>
</tr>
<tr>
<td>NumberOfRemoteUDPServerSocket</td>
<td>Required; the default value for this property is 4. The NumberOfRemoteUDPServerSockets property allows you to configure the number of source ports used while proxying requests to a remote RADIUS server. If the NumberOfRemoteUDPServerSockets property is set to a value ( n ), all remote servers share and use ( n ) sockets. The NumberOfRemoteUDPServerSockets value comprises a number, as in ( n ), where ( n ) should be less than or equal to the current process file descriptor limit divided by 4. Note By default, the RADIUS process supports up to 1024 file descriptors. To increase the file descriptors, stop the arserver; in the arserver script, specify the required value to “NUMBER_OF_FILE_DESCRIPTORS” and restart the server. The value for “NUMBER_OF_FILE_DESCRIPTORS” should be in the range between 1024 to 65535.</td>
</tr>
<tr>
<td>MemoryLimitForRadiusProcess</td>
<td>This property is used to avoid crashing of the RADIUS process. The default value is 3500 Megabytes. This property is under /radius/advanced. When the RADIUS process uses memory more than the configured limit, further sessions are not created and Prime Access Registrar rejects further incoming requests.</td>
</tr>
<tr>
<td>MemorySizeCheckInterval</td>
<td>This property is used to avoid crashing of the RADIUS process. This is used in conjunction with MemoryLimitForRadiusProcess. The default value is 5 minutes. MemorySizeCheckInterval is a hidden parameter in mcd database. To modify the default value, you need to export the mcd database. Typically, a separate thread is created to monitor the RADIUS process memory usage for every 5 minutes.</td>
</tr>
<tr>
<td>UDPPacketSize</td>
<td>Required; the default is 4096. RFC 2138 specifies the maximum packet length can be 4096 bytes. Do not change this value.</td>
</tr>
</tbody>
</table>
Chapter 3  Using the Graphical User Interface

Configuring Cisco Prime Access Registrar

PerPacketHeapSize Required; the default is 6500. This property sets the size of the initial heap for each packet. The heap is the dynamic memory a request can use during its lifetime. By preallocating the heap size at the beginning of request processing, we can minimize the cost of memory allocations. If PerPacketHeapSize is too low, Prime Access Registrar will ask the system for memory more often. If PerPacketHeapSize is too high, Prime Access Registrar will allocate too much memory for the request causing the system to use more memory than required.

MinimumSocketBufferSize Required; the default is 65536 (64 K). This property governs how deep the system’s buffer size is for queueing UDP datagrams until Cisco Prime Access Registrar can read and process them. The default is probably sufficient for most sites. You can, however, raise or lower it as necessary.

MaximumOutstandingRequests Optional; the default value for this property is 0. The MaximumOutstandingRequests property is used to limit the incoming traffic in terms of “requests processed”. Serves as a hard limit.

The MaximumOutstandingRequests property comprises a number \( n \), where \( n \) can be any nonzero value.

MaximumIncomingRequests Optional; the default value for this property is 0.

ARIsCaseInsensitive When set to FALSE, requires that you provide exact pathnames with regard to upper and lower case for all objects, subobjects, and properties. The default setting, TRUE, allows you to enter paths such as /rad/serv instead of /Rad/Serv.

Note  Prime Access Registrar always authenticates the RADIUS attribute User-Name with regard to upper and lower case, regardless of the setting of this flag.

EnableDiameter Optional; Either TRUE or FALSE; default is TRUE. Set to True when you want to use the Diameter protocol in Prime Access Registrar.

Table 3-33  BackingStore/ServerParameter Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PerPacketHeapSize</td>
<td>Required; the default is 6500. This property sets the size of the initial heap for each packet. The heap is the dynamic memory a request can use during its lifetime. By preallocating the heap size at the beginning of request processing, we can minimize the cost of memory allocations. If PerPacketHeapSize is too low, Prime Access Registrar will ask the system for memory more often. If PerPacketHeapSize is too high, Prime Access Registrar will allocate too much memory for the request causing the system to use more memory than required.</td>
</tr>
<tr>
<td>MinimumSocketBufferSize</td>
<td>Required; the default is 65536 (64 K). This property governs how deep the system’s buffer size is for queueing UDP datagrams until Cisco Prime Access Registrar can read and process them. The default is probably sufficient for most sites. You can, however, raise or lower it as necessary.</td>
</tr>
<tr>
<td>MaximumOutstandingRequests</td>
<td>Optional; the default value for this property is 0. The MaximumOutstandingRequests property is used to limit the incoming traffic in terms of “requests processed”. Serves as a hard limit. The MaximumOutstandingRequests property comprises a number ( n ), where ( n ) can be any nonzero value.</td>
</tr>
<tr>
<td>MaximumIncomingRequests</td>
<td>Optional; the default value for this property is 0.</td>
</tr>
<tr>
<td>ARIsCaseInsensitive</td>
<td>When set to FALSE, requires that you provide exact pathnames with regard to upper and lower case for all objects, subobjects, and properties. The default setting, TRUE, allows you to enter paths such as /rad/serv instead of /Rad/Serv.</td>
</tr>
<tr>
<td>EnableDiameter</td>
<td>Optional; Either TRUE or FALSE; default is TRUE. Set to True when you want to use the Diameter protocol in Prime Access Registrar.</td>
</tr>
</tbody>
</table>
Chapter 3 Using the Graphical User Interface

Configuring Cisco Prime Access Registrar 7.0.1 User Guide

Table 3-33 BackingStore/ServerParameter Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeyStores</td>
<td>This section is available for each of the following EAP services:</td>
</tr>
<tr>
<td></td>
<td>● EAP-SIM</td>
</tr>
<tr>
<td></td>
<td>● EAP-SIM-3GPP</td>
</tr>
<tr>
<td></td>
<td>● EAP-AKA</td>
</tr>
<tr>
<td></td>
<td>● EAP-AKA-3GPP</td>
</tr>
<tr>
<td></td>
<td>● EAP-AKA-PRIME</td>
</tr>
<tr>
<td></td>
<td>● EAP-AKA-PRIME-3GPP</td>
</tr>
<tr>
<td></td>
<td>● EAP-FAST</td>
</tr>
<tr>
<td>NumberOfKeys</td>
<td>Maximum number of keys stored for generating pseudonym secrets. Value can be from 1 till 1024.</td>
</tr>
<tr>
<td>RolloverPeriod</td>
<td>Duration between key updates. Default is 1 week.</td>
</tr>
<tr>
<td></td>
<td>In case of rolling encryption, this denotes the duration for which a key is active, after which the key is expired and the next key is considered as an active key for pseudonym generation. For more information on rolling encryption, see Rolling Encryption Support for Pseudonym Generation in EAP-SIM, EAP-AKA, and EAP-AKA’ Services, page 9-60</td>
</tr>
</tbody>
</table>

Setting Server Parameters

To set up new server parameters:

Step 1 Choose Configuration > Advanced > Backing/ServerParam. The Backing/ServerParam Advanced Details page is displayed.

Step 2 Specify the relevant details.

Step 3 Click Set to save the specified details in the Backing/ServerParamAdvanced Details page.

On successful creation of the server parameters, a success message is displayed else a respective error message is displayed.

RemoteSessionServer

Prime Access Registrar sessions can also be stored on a remote database. This improves the overall scalability of the number of sessions that Prime Access Registrar can simultaneously handle.

The remote session manager internally uses the following two ODBC remote servers:

● Internal-ODBC-Read-Server
● Internal-ODBC-Write-Server
Configurations pertaining to these internal remote servers can be done under the RemoteSessionServer section.

**Note**
Ensure that the length of fields such as Username, Session/Resource Manager name Session-Key, Query-Key and so on are limited to the value specified in the schema, while it is configured. Although the field length of entire session record is 3KB it is limited to 2KB. This is practically sufficient to hold all the session parameters as well as the cached attributes (if any). For more information about the schema, see Remote Session Management, page 17-47.

**Note**
Remote session manager will work only with Oracle database.

Table 3-34 lists and describes the fields in the RemoteSessionServer Advanced Details page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReactivateTimerInterval</td>
<td>Mandatory time interval (in milliseconds) to activate an inactive server; defaults to 300000 ms.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Mandatory time interval (in seconds) to wait for SQL operation to complete; defaults to 15 seconds</td>
</tr>
<tr>
<td>DataSourceConnections</td>
<td>Mandatory number of connections to be established; defaults to 8</td>
</tr>
<tr>
<td>ODBCDataSource</td>
<td>Name of the ODBCDataSource to use and must refer to one entry in the list of ODBC datasources configured under /Radius/Advanced/ODBCDataSources. Mandatory; no default.</td>
</tr>
<tr>
<td>KeepAliveTimerInterval</td>
<td>Mandatory time interval (in milliseconds) to send a keepalive to keep the idle connection active; defaults to zero (0) meaning the option is disabled</td>
</tr>
<tr>
<td>MaximumBufferFileSize</td>
<td>Mandatory if BufferAccountingPackets is set to TRUE, determines the maximum buffer file size, defaults to 10 Megabyte)</td>
</tr>
<tr>
<td>CacheLimit</td>
<td>Default is 250000; This represents the overall limit on cache of all 'remote' session managers. This value is interpreted as the maximum number of packets that can be present in cache. When the number of sessions hits this limit, sessions will be 'cached out'. This cache out operation will continue, until the cache is at least 20% free.</td>
</tr>
</tbody>
</table>
Setting RemoteSessionServer Details

To set a new RemoteSessionServer details:


Step 2  Specify the relevant details.

Step 3  Click Set to save the specified details in the RemoteSessionServer Advanced Details page.

On successful creation of the RemoteSessionServer details, a success message is displayed else a respective error message is displayed.

SNMP

Prime Access Registrar provides SNMP MIB for users of network management systems. The supported MIBs enable the network management station to collect state and statistic information from a Prime Access Registrar server. It enables a standard SNMP management station to check the current state of the server as well as the statistics on each client or each proxy remote server. These messages contain information indicating that either the server was brought up or down or that the proxy remote server is down or has come back online.

Table 3-35 lists and describes the fields in the SNMP Advanced Details page.
### Table 3-35  SNMP Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SNMP Info section</strong></td>
<td></td>
</tr>
<tr>
<td>InputQueueHighThreshold</td>
<td>Percentage that indicates the upper limit of the packet input queue usage. Default is 90. Prime Access Registrar supports traps to indicate input queue usage. When the input buffer exceeds the given high threshold value, Prime Access Registrar generates a carInputQueueFull trap.</td>
</tr>
<tr>
<td>InputQueueLowThreshold</td>
<td>Percentage that indicates the lower limit of the packet input queue usage. Default is 60. After reaching the high threshold, if the buffer usage drops below a low threshold value, Prime Access Registrar generates a carInputQueueNotVeryFull trap.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Check the box to enable SNMP settings.</td>
</tr>
<tr>
<td>TracingEnabled</td>
<td>Check the box to enable all possible tracing in SNMP agent. Tracing is used for debugging purposes.</td>
</tr>
<tr>
<td>MasterAgentEnabled</td>
<td>To use SNMP, enable the master agent. Prime Access Registrar responds to SNMP queries through the SNMP master agent.</td>
</tr>
<tr>
<td><strong>RFC Compliance Info section</strong></td>
<td></td>
</tr>
<tr>
<td>AllowRejectAttrs</td>
<td>When AllowRejectAttrs is set to FALSE, Reply-Message attributes will not be passed in an Access Reject packet. When AllowRejectAttrs is set to TRUE, attributes will be allowed to pass in an Access Reject packet.</td>
</tr>
<tr>
<td>AllowEAPRejectAttrs</td>
<td>When AllowEAPRejectAttrs is set to FALSE, Reply-Message attributes will not be passed in an Access Reject packet if the packet contains EAP-Message attribute. When AllowEAPRejectAttrs is set to TRUE, attributes will be allowed to pass in an Access Reject packet even if the packet contains EAP-Message attribute.</td>
</tr>
<tr>
<td><strong>Reply Messages section</strong></td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends this value when the property corresponding to the reject reason is not set.</td>
</tr>
<tr>
<td>UnknownUser</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the Reply-Message attribute whenever Cisco Prime Access Registrar cannot find the user specified by User-Name.</td>
</tr>
<tr>
<td>UserNotEnabled</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the Reply-Message attribute whenever the user account is disabled.</td>
</tr>
<tr>
<td>UserPasswordInvalid</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the Reply-Message attribute whenever the password in the Access-Request packet did not match the password in the database.</td>
</tr>
</tbody>
</table>
### Table 3-35  SNMP Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UnableToAcquireResource</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <code>Reply-Message</code> attribute whenever one of the Resource Managers was unable to allocate the resource for this request.</td>
</tr>
<tr>
<td>ServiceUnavailable</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <code>Reply-Message</code> attribute whenever a service the request needs (such as a RemoteServer) is unavailable.</td>
</tr>
<tr>
<td>InternalError</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <code>Reply-Message</code> attribute whenever an internal error caused the request to be rejected.</td>
</tr>
<tr>
<td>MalformedRequest</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <code>Reply-Message</code> attribute whenever a required attribute (such as <code>User-Name</code>) is missing from the request.</td>
</tr>
<tr>
<td>ConfigurationError</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <code>Reply-Message</code> attribute whenever the request is rejected due to a configuration error. For example, if a script sets an environment variable to the name of an object such as <code>Authentication-Service</code>, and that object does not exist in the configuration, the reason reported is <code>ConfigurationError</code>.</td>
</tr>
<tr>
<td>IncomingScriptFailed</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <code>Reply-Message</code> attribute whenever one of the <code>IncomingScripts</code> fails to execute.</td>
</tr>
<tr>
<td>OutgoingScriptFailed</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <code>Reply-Message</code> attribute whenever one of the <code>OutgoingScripts</code> fails to execute.</td>
</tr>
<tr>
<td>IncomingScriptRejectedRequest</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <code>Reply-Message</code> attribute whenever one of the <code>IncomingScripts</code> rejects the Access-Request.</td>
</tr>
<tr>
<td>TerminationAction</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <code>Reply-Message</code> attribute whenever Cisco Prime Access Registrar processes the Access-Request as a Termination-Action and is being rejected as a safety precaution.</td>
</tr>
<tr>
<td>OutgoingScriptRejectedRequest</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <code>Reply-Message</code> attribute whenever one of the <code>OutgoingScripts</code> rejects the Access-Request.</td>
</tr>
</tbody>
</table>

### Setting SNMP Details

To set up new SNMP details:

**Step 1**  Choose **Configuration > Advanced > SNMP**. The SNMP Advanced Details page is displayed.

**Step 2**  Specify the relevant details.
Click **Set** to save the specified details in the SNMP Advanced Details page. On successful creation of the SNMP details, a success message is displayed else a respective error message is displayed.

### DDNS

Prime Access Registrar supports Dynamic DNS Remote server. It is a method, protocol, or network that notifies the server to change the active DNS configuration of its configured hostnames, addresses or other information stored in DNS.

You can click the Add button in the DDNS Details page to enter the TSIGKeys details in the TSIGKeys Details section.

Table 3-36 lists and describes the fields in the TSIGKeys Details section.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the TSIG Key.</td>
</tr>
<tr>
<td>Secret</td>
<td>Set to the same base64-encoded string as defined in the DNS server.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the TSIG Key</td>
</tr>
</tbody>
</table>

You can use the DDNS Details page for the following:

- Filtering Records
- Setting DDNS Details
- Adding the TSIGKeys for DDNS
- Editing Records
- Deleting Records

### Setting DDNS Details

To set up new DDNS details:

**Step 1** Choose **Configuration > Advanced > DDNS**. The DDNS Details page is displayed.

**Step 2** Check the **SynthesizeReverseZone** check box, and click **Set DDNS**.

### Adding the TSIGKeys for DDNS

To add TSIGKeys details for DDNS:

**Step 1** Choose **Configuration > Advanced > DDNS**. The DDNS Details page is displayed.

**Step 2** Click **Add**. The TSIGKeys details section is displayed.

**Step 3** Enter the relevant details.

**Step 4** Click **Add** to save the specified details in the TSIGKeys Details section.
On successful creation of the TSIGKeys details, a success page is displayed else a respective error message is displayed.

**ODBC DataSources**

Prime Access Registrar uses ODBC as the datasource name to be used by the remote server. Multiple remote servers can use the same ODBCDataSource. Under the ODBCDataSource object definition, a list defines ODBC.ini filename/value pairs for a connection. The list includes a Type field and a Driver field, different for each Driver and Data Source, to indicate its Driver and Data Source. Prime Access Registrar supports only the Easysoft Open Source Oracle Driver.

Table 3-37 lists and describes the fields in the Add ODBC DataSources page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the ODBCDataSource</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; Description of the ODBC Data Source</td>
</tr>
<tr>
<td>Type</td>
<td>Required; type of the ODBC data source, which could be myodbc or oracle_oci.</td>
</tr>
<tr>
<td>Driver</td>
<td>Required; liboardodbc.so (default value)</td>
</tr>
<tr>
<td>UserID</td>
<td>Required; database username (no default value)</td>
</tr>
<tr>
<td>Password</td>
<td>Optional; user password; shown encrypted</td>
</tr>
<tr>
<td>DataBase</td>
<td>Required; Oracle Client configuration database name (no default value)</td>
</tr>
<tr>
<td>Server</td>
<td>Set the name of the server</td>
</tr>
<tr>
<td>Port</td>
<td>Set the port details.</td>
</tr>
</tbody>
</table>

You can use the ODBC DataSources page for the following:

- Filtering Records
- Adding ODBC Data Source
- Log
- Editing Records
- Deleting Records

**Adding ODBC Data Source**

To add new ODBC data source details:

**Step 1** Choose Configuration > Advanced > ODBC DataSources. The ODBC DataSources page is displayed.

**Step 2** Click Add to add new ODBC data source details. The ODBC DataSources Details page is displayed.

**Step 3** Enter the relevant details.
Step 4  Click **Submit** to save the specified details. Otherwise click **Cancel** to return to the ODBC DataSources page without saving the details.

The ODBC DataSources page is displayed with the newly added details and a success message is displayed else a respective error message is displayed.

---

**Log**

The log files defined in Prime Access Registrar assist you in identifying the issues related to it. Prime Access Registrar holds sets of log files to store information relevant to server agent processes, monitoring arserver utility, execution of aregcm command, mcd internal database details, RADIUS server processes and debug details of RADIUS request process.

Table 3-38 lists and describes the fields in the Log Files page.

**Table 3-38 Log Details**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>GUI Log Settings section</strong></td>
<td></td>
</tr>
<tr>
<td>LOG LEVEL</td>
<td>Select either Debug level or Error.</td>
</tr>
<tr>
<td>MaxFileSize</td>
<td>Set the maximum size of the log file.</td>
</tr>
<tr>
<td><strong>Advance Details section</strong></td>
<td></td>
</tr>
<tr>
<td>LogFileSize</td>
<td>Required; the default is 1 megabyte. This property specifies the maximum size of the RADIUS server log file. The value for the LogFileSize field is a string composed of two parts; a number, and a units indicator (&lt;n&gt; &lt;units&gt;) in which the unit is one of: K, kilobyte, kilobytes, M, megabyte, megabytes, G, gigabyte, or gigabytes. The LogFileSize property does not apply to the config_mcd_1_log or agent_server_1_log files.</td>
</tr>
<tr>
<td>LogFileCount</td>
<td>Required; the default is 2. This property specifies the number of log files to be kept on the system. A new log file is created when the log file size reaches LogFileCount. The LogFileCount property does not apply to the config_mcd_1_log or agent_server_1_log files.</td>
</tr>
<tr>
<td>TraceFileSize</td>
<td>Required; the default is 1 GB. This property specifies the size of the trace files to be kept on the system. A new trace file is created when the trace file size reaches TraceFileSize. The value for the TraceFileSize field is a string composed of two parts; a number, and a units indicator (&lt;n&gt; &lt;units&gt;) in which the unit is one of: K, kilobyte, kilobytes, M, megabyte, megabytes, G, gigabyte, or gigabytes.</td>
</tr>
<tr>
<td>TraceFileCount</td>
<td>Required; this value can be set from 1–100, and the default is 2. This property specifies the number of trace files to maintain. A value of 1 indicates that no file rolling occurs.</td>
</tr>
</tbody>
</table>

---

Log files are essential for diagnosing issues, monitoring server activity, and understanding the operational status of the system. Properly configured log settings ensure that critical events are captured and can be easily reviewed.
### Table 3-38  Log Details (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LogServerActivity</td>
<td>Required; the default is FALSE, which means Cisco Prime Access Registrar logs all responses except Access-Accepts and Access-Challenges. Accepting the default reduces the load on the server by reducing that amount of information it must log. Note, the client is probably sending accounting requests to an accounting server, so the Access-Accept requests are being indirectly logged. When you set it to TRUE, Cisco Prime Access Registrar logs all responses to the server log file.</td>
</tr>
<tr>
<td>TraceLevel</td>
<td>Set the trace level.</td>
</tr>
<tr>
<td>LogTPSActivity</td>
<td>When set to TRUE, this property enables to log the TPS usage in a CSV file. The TPS is logged in the following format: <code>&lt;mm-dd-yyyy&gt;, &lt;hh:mm:ss&gt;, &lt;tps-value&gt;</code> For example, 04-01-2013, 12:00:01, 102 The default is False.</td>
</tr>
<tr>
<td>TPSLogFileCount</td>
<td>Required only if you check the LogTPSActivity check box; the number of TPS Sampling log files to maintain in the repository. The default value is 2.</td>
</tr>
<tr>
<td>TPSLogFileNamePrefix</td>
<td>Required only if you check the LogTPSActivity check box; this represents the prefix of the CSV file which will be available in the logs directory of Prime Access Registrar. The following represents the CSV filename format: <code>&lt;user-prefix&gt;-&lt;mm-dd-yyyy&gt;.csv</code> tps-04-01-2013.csv</td>
</tr>
<tr>
<td>TPSSamplingPeriodInSecs</td>
<td>Required only if you check the LogTPSActivity check box; this represents the TPS sampling period in seconds. The minimum sampling period is set to 5. The default is 30.</td>
</tr>
<tr>
<td>EnableSIGTRANStackLogs</td>
<td>When set to TRUE, this property enables to log the SIGTRAN stack logs in stack.log file.</td>
</tr>
<tr>
<td>SIGTRANLogFileSize</td>
<td>Required if you check the EnableSIGTRANStackLogs check box. This property specifies the maximum size (in megabyte) of the SIGTRAN stack log file.</td>
</tr>
<tr>
<td>SIGTRANLogFileCount</td>
<td>Required if you check the EnableSIGTRANStackLogs check box. This value can be set from 1–100, and the default is 10. This property specifies the number of SIGTRAN log files to maintain in the repository.</td>
</tr>
<tr>
<td>LogSessionActivity</td>
<td>When set to TRUE, this property enables Prime Access Registrar to log the session count in the server.</td>
</tr>
<tr>
<td>SessionLogFileCount</td>
<td>Required only if you check the LogSessionActivity check box; the number of session log files to maintain in the repository. The default value is 2.</td>
</tr>
</tbody>
</table>
You can use the Log Files page for the following:

- Filtering Records
- Viewing Log Details
- Downloading Log Details
- Setting Log Details

### Viewing Log Details

To view the log files:

**Step 1** Choose Configuration > Advanced > Log. The Log Files page is displayed.

**Step 2** Choose the appropriate radio button and click View to view the file.

### Downloading Log Details

To download the log files:

**Step 1** Choose Configuration > Advanced > Log. The Log Files page is displayed.

**Step 2** Choose the appropriate radio button and click Download to download the file.

### Setting Log Details

To set the log details:

**Step 1** Choose Configuration > Advanced > Log. The Log Files page is displayed.

**Step 2** Enter the relevant details and click Set to save the specified details.

### Ports

The Ports list specifies which ports to listen to for requests. When you specify a port, Prime Access Registrar makes no distinction between the port used to receive Access-Requests and the port used to receive Accounting-Requests. Either request can come in on either port.
Most NASs send Access-Requests to port 1812 and Accounting-Requests to 1813, however, Prime Access Registrar does not check.

When you do not specify any ports, Prime Access Registrar reads the /etc/services file for the ports to use for access and accounting requests. If none are defined, Prime Access Registrar uses the standard ports (1812 and 1813).

Table 3-39 lists and describes the fields in the Ports page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
<td>Required; allows you to use ports other than the default, 1812 and 1813. You can use this option to configure Prime Access Registrar to use other ports. If you add additional ports, however, Prime Access Registrar will use the added ports and no longer use the default ports 1812 and 1813. These default ports can still be used by adding them to the list of ports to use.</td>
</tr>
<tr>
<td>Type</td>
<td>Set the port type.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the port.</td>
</tr>
</tbody>
</table>

You can use the Ports page for the following:
- Filtering Records
- Adding Port Details
- Interfaces
- Editing Records
- Deleting Records

### Adding Port Details

To add new port details:

1. **Step 1** Choose **Configuration > Advanced > Port**. The Ports page is displayed.
2. **Step 2** Enter the relevant details and click **Add**. The new port details will be listed in the Ports page.

### Interfaces

The Interfaces list specifies the interfaces on which the RADIUS server receives and sends requests. You specify an interface by its IP address.
- When you set an IP address, Prime Access Registrar uses that interface to send and receive Access-Requests.
- When no interfaces are listed, the server performs an interface discover and uses all interfaces of the server, physical and logical (virtual).

**Note**
The IP address format is enhanced to support both IPv4 and IPv6.
You can use the interfaces page for the following:

- Filtering Records
- Adding IP Addressing Interface
- Deleting Records

**Adding IP Addressing Interface**

To add a new IP address interface to define an interface:

**Step 1** Choose Configuration > Advanced > Interfaces. The Interfaces page is displayed.

**Step 2** Enter the IP Address and click Add.

The Interfaces page is displayed with the newly added details and a success message is displayed else a respective error message is displayed.

**Attribute Groups**

The Attributes can be grouped using Prime Access Registrar Profile object. The attributes for a particular user group can be grouped under a profile and the attributes contained in the profiles will be returned in their access-accepts.

Table 3-40 lists and describes the fields in the Attribute Groups Details page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the attribute group.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the attribute group.</td>
</tr>
<tr>
<td>Attribute type</td>
<td>Select either RADIUS or VENDOR. If Vendor is selected, specify the vendor type from the drop-down list.</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Optional; based on the Attribute Type selected, the attribute name is automated. Set the relevant name for the attribute type selected. Click the Add button to save the details and list it in Attribute list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
</tbody>
</table>

You can use the Attribute Groups page for the following:

- Filtering Records
- Adding Attribute Group Details
- Rules
- Editing Records
- Deleting Records
Adding Attribute Group Details

To add new attribute groups details:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Choose Configuration &gt; Advanced &gt; Attributes Groups. The Attribute Groups page is displayed.</td>
</tr>
<tr>
<td>Step 2</td>
<td>Click Add to add new attribute group details. The Attribute Group Details page is displayed.</td>
</tr>
<tr>
<td>Step 3</td>
<td>Enter the relevant details.</td>
</tr>
<tr>
<td>Step 4</td>
<td>Click Submit to save the specified details in the Attribute Groups Details page. Otherwise click Cancel to return to the Attribute Groups page without saving the details.</td>
</tr>
</tbody>
</table>

The Attribute Groups page is displayed with the newly added details or a respective error message is displayed.

Rules

A Rule is a function that selects services based on all input information used by the function. Table 3-41 lists and describes the fields in the Add Rules List page.

<table>
<thead>
<tr>
<th>Table 3-41</th>
<th>Rule Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fields</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td><strong>General Properties tab</strong></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Required; must be unique in the Rule list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the rule.</td>
</tr>
<tr>
<td>Type</td>
<td>Required; specifies the type of the rule which can be Radius or Diameter.</td>
</tr>
<tr>
<td>Script Name</td>
<td>Name of the script.</td>
</tr>
<tr>
<td><strong>Attribute Details tab</strong></td>
<td></td>
</tr>
<tr>
<td>RADIUS</td>
<td>Optional; set Radius, if the attribute and value need to be defined for RADIUS.</td>
</tr>
<tr>
<td>VENDOR</td>
<td>Optional; set Vendor, if the attribute and value need to be defined for Vendor.</td>
</tr>
<tr>
<td>AttributeName</td>
<td>Optional; based on the Attribute Type selected, the attribute name is automated. Set the relevant name for the attribute type selected.</td>
</tr>
<tr>
<td>AttributeValue</td>
<td>Optional; set the value for the selected attribute. Click the Add button to save the details and list it in Name and Value list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
</tbody>
</table>

You can use the Rules List page for the following:
Configuring Cisco Prime Access Registrar

Chapter 3      Using the Graphical User Interface

- Filtering Records
- Setting Rules
- Session Managers
- Editing Records
- Deleting Records

Setting Rules

To set new rules:

Step 1  Choose Configuration > Rules. The List of Rules page is displayed.

Step 2  Click Add. The Rules Details page is displayed.

Step 3  Enter the relevant details.

Step 4  Click Submit to save the specified details in the Rules Details page. Otherwise click Cancel to return to the List of Rules page without saving the details.

The List of Rules page is displayed with the newly added details or a respective error message is displayed.

Session Managers

You can use Session Managers to track user sessions. The Session Managers monitor the flow of requests from each NAS and detect the session state. When requests come through to the Session Manager, it creates sessions, allocates resources from appropriate Resource Managers, and frees and deletes sessions when users log out.

The Session Manager enables you to allocate dynamic resources to users for the lifetime of their session. You can define one or more Session Managers and have each one manage the sessions for a particular group or company.

Note  Session record size is limited by the operating system (OS) paging size (4 KB in Linux). If a request triggers creation of a session that exceeds the OS paging size, the request will be dropped and the session will not be created.

Note  In this release of Prime Access Registrar, the memory capacity is enhanced to store more than 4 million active session’s by storing the active session records in database server instead of storing it in the main memory. The capacity is dependent on the number of attributes that are being captured for each session.
If the disk partition where Prime Access Registrar stores session backing store data (usually the disk partition where Prime Access Registrar is installed, such as /opt/CSCOar) is full, the subsequent packets that try to create sessions will be dropped and no sessions will be created due to lack of disk space.

Session Managers use Resource Managers, which in turn, manage a pool of resources of a particular type.

Table 3-42 lists and describes the fields in the Session Manager Details page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Session Managers list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the Session Manager.</td>
</tr>
<tr>
<td>Type</td>
<td>Required; set to local or remote. Local is the traditional session manager that maintains sessions in memory and has good performance. The remote session manager operates on a remote ODBC database, and its performance is highly dependent on the performance of the ODBC database.</td>
</tr>
<tr>
<td>EnableDiameter</td>
<td>Optional; check the box if you want to use the session manager for Diameter services.</td>
</tr>
</tbody>
</table>
| SessionKey    | SessionKey property is used to set the sessionkey value for the Session Manager. The SessionManager checks whether the environmental variable Session-Key is set or not. If the environmental variable is set, the server uses it as the sessionkey. If environmental variable Session-Key is not set then SessionManager gets the value configured in the SessionKey property under SessionManager. SessionKey can be a combination of attributes separated by a colon. The values for those attributes are obtained from the RequestDictionary. If any one of the attribute that is configured for the sessionkey is not present in the RequestDictionary, Prime Access Registrar will drop the request. However, if Session-Key is not set, SessionManager uses NAS-Identifier and NAS-Port to create the sessionkey. An example configuration, 

```bash
--> set SessionKey "User-Name:NAS-Port"
```

The following shows the sample configuration of sessionkey for Session Manager:

```bash
[ //localhost/Radius/SessionManagers/session-mgr-1 ]
Name = session-mgr-1
Description =
Type = local
EnableDiameter = FALSE
IncomingScript =
OutgoingScript =
AllowAccountingStartToCreateSession = TRUE
SessionTimeOut =
PhantomSessionTimeOut =
SessionKey =
ResourceManagers/
```
### Table 3-42  Session Manager Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowAccountingStartToCreateSession</td>
<td>Set to TRUE by default; start the session when the Prime Access Registrar server receives an Access Accept or an Accounting-Start. When set to FALSE, start the session when the Prime Access Registrar server receives an Access Accept.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; name of script to run when the service starts. This script is run as soon as the session is acquired in Prime Access Registrar.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; script to be run just before the session is written to backing store.</td>
</tr>
<tr>
<td>SessionTimeOut</td>
<td>The SessionTimeOut property is optional; no value for this property means the session timeout feature is disabled. Used in conjunction with <code>/Radius/Advanced/SessionPurgeInterval</code> for the session timeout feature. Enables the session timeout feature for a Session Manager. If the SessionTimeOut property is set to a value under a session manager, all sessions that belong to that session manager will be checked for timeouts at each SessionPurgeInterval. If any sessions have timed out, they will be released, and all resources associated with those sessions are also released. The SessionTimeOut property determines the timeout for a session. If the time difference between the current time and the last update time is greater than this property’s value, the session is considered to be stale. The last update time of the session is the time at which the session was created or updated. The SessionTimeOut value is comprised of a number and a units indicator, as in ( n ) units, where a unit is one of minutes, hours, days, or weeks. The default unit is ‘days’.</td>
</tr>
<tr>
<td>PhantomSessionTimeOut</td>
<td>Optional; no value for this property means the phantom session timeout feature is disabled. The PhantomSessionTimeOut property is used in conjunction with <code>/Radius/Advanced/SessionPurgeInterval</code> to enable the phantom session timeout feature for Session Manager. If the PhantomSessionTimeOut property is set to a value under a session manager, all sessions that belong to that session manager will be checked for receipt of an Accounting-Start packet. Sessions that do not receive an Accounting-Start packet from creation until its timeout will be released. The PhantomSessionTimeOut value comprises a number and a units indicator, as in ( n ) units, where a unit is one of minutes, hours, days, or weeks. The default unit is ‘days’.</td>
</tr>
<tr>
<td>SessionCreationCmdList</td>
<td>Available only if you check the EnableDiameter check box; session created for the configured application, command code, and AVP.</td>
</tr>
</tbody>
</table>
You can use the Session Managers page for the following:

- Filtering Records
- Adding Session Manager Details
- Editing Records
- Deleting Records

### Adding Session Manager Details

To add new session manager details:

**Step 1** Choose **Configuration > Session Managers**. The Session Managers page is displayed.

**Step 2** Click **Add**. The Session Manager Details page is displayed.

**Step 3** Enter the required details.

**Step 4** Click **Add** to save the specified details in the Session Manager Details page. Otherwise click **Cancel** to return to the Session Managers page without saving the details.

The Session Managers page is displayed with the newly added details or a respective error message is displayed.

---

### ResourceManager

ResourceManager

Resource Managers allow you to allocate dynamic resources to user sessions. The following lists the different types of Resource Managers.

- **IP-Dynamic**—manages a pool of IP addresses that allows you to dynamically allocate IP addresses from a pool of addresses
- **IP-Per-NAS-Port**—allows you to associate ports to specific IP addresses, and thus ensure each NAS port always gets the same IP address
- **IPX-Dynamic**—manages a pool of IPX network addresses
- **Subnet-Dynamic**—manages a pool of subnet addresses
- **Group-Session-Limit**—manages concurrent sessions for a group of users; that is, it keeps track of how many sessions are active and denies new sessions after the configured limit has been reached
• **User-Session-Limit**—manages per-user concurrent sessions; that is, it keeps track of how many sessions each user has and denies the user a new session after the configured limit has been reached.

• **Home-Agent**—manages a pool of on-demand IP addresses.

• **USR-VPN**—manages Virtual Private Networks (VPNs) that use USR NAS Clients.

• **Home-Agent-IPv6**—manages a pool of on-demand IPv6 addresses.

• **Remote-IP-Dynamic**—manages a pool of IP addresses that allows you to dynamically allocate IP addresses from a pool of addresses. It internally works with a remote ODBC database.

• **Remote-User-Session-Limit**—manages per-user concurrent sessions; that is, it keeps track of how many sessions each user has and denies the user a new session after the configured limit has been reached. It internally works with a remote ODBC database.

• **Remote-Group-Session-Limit**—manages concurrent sessions for a group of users; that is, it keeps track of how many sessions are active and denies new sessions after the configured limit has been reached. It internally works with a remote ODBC database.

• **Session Cache**—allows you to define the RADIUS attributes to store in cache.

• **Dynamic-DNS**—manages the DNS server.

• **Remote-Session-Cache**—allows you to define the RADIUS attributes to store in cache. It should be used with session manager of type ‘remote’.

• **3GPP**—allows you to define the attribute for 3GPP authorization.

Each Resource Manager is responsible for examining the request and deciding whether to allocate a resource for the user, do nothing, or cause Cisco Prime Access Registrar to reject the request.

Table 3-43 lists and describes the fields in the Resource Manager Details page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource Manager Name</td>
<td>Required; must be unique in the Resource Managers list.</td>
</tr>
<tr>
<td>Description (optional)</td>
<td>Optional; description of the Resource Manager.</td>
</tr>
<tr>
<td>Type</td>
<td>Required; must be either Dynamic-DNS, IP-Dynamic, IP-Per-NAS-Port, IPX-Dynamic, Session Cache, Subnet-Dynamic, Group-Session-Limit, Home-Agent, User-Session-Limit, USR-VPN, Home-Agent-IPv6, Remote-IP-Dynamic, Remote-User-Session-Limit, Remote-Group-Session-Limit, Remote-Session-Cache, or 3GPP. Based on the option selected, the fields displayed in the Resource Manager Details page varies.</td>
</tr>
</tbody>
</table>

The fields displayed in the Resource Manager Details page changes based on the option selected in the Type field. The following tables describe the fields in the Resource Manager Details page.

**DYNAMIC-DNS**

Table 3-44 lists and describes the fields in the Resource Manager Details page.
### Table 3-44 DYNAMIC-DNS Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General tab</td>
<td></td>
</tr>
<tr>
<td>Max DNS TTLS</td>
<td>Set the maximum TTL of the DNS record.</td>
</tr>
<tr>
<td>DNS Host bytes</td>
<td>Set the number of bytes to be used to construct the reverse zone entry.</td>
</tr>
<tr>
<td>Forward Zone Name</td>
<td>Set the name of the forward zone. For a given Resource Manager you must decide which forward zone you will be updating for sessions the resource manager will manage.</td>
</tr>
<tr>
<td>Reverse Zone Name</td>
<td>Set the name of the reverse zone.</td>
</tr>
<tr>
<td>Forward Zone Server</td>
<td>Set the Server IP of the forward zone.</td>
</tr>
<tr>
<td>Reverse Zone Server</td>
<td>Set the Server IP of the reverse zone.</td>
</tr>
<tr>
<td>Forward Zone TSIG Keys</td>
<td>Server-wide security key to process all forward zone dynamic DNS updates. This is used if a ForwardZoneTSIGKey was not specified on the Resource Manager.</td>
</tr>
<tr>
<td>Reverse Zone TSIG Keys</td>
<td>Server-wide security key to process all reverse zone dynamic DNS updates. This is used if a ReverseZoneTSIGKey was not specified on the Resource Manager.</td>
</tr>
</tbody>
</table>

### GROUP-SESSION-LIMIT

Table 3-45 lists and describes the fields in the Resource Manager Details page.

#### Table 3-45 GROUP-SESSION-LIMIT Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Session Limit</td>
<td>Set the GroupSessionLimit property to the maximum number of concurrent sessions for all users.</td>
</tr>
</tbody>
</table>

### REMOTE-GROUP-SESSION-LIMIT

Table 3-46 lists and describes the fields in the Resource Manager Details page.

#### Table 3-46 REMOTE-GROUP-SESSION-LIMIT Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Session Limit</td>
<td>Set the GroupSessionLimit property to the maximum number of concurrent sessions for all users.</td>
</tr>
</tbody>
</table>

### HOME-AGENT

Table 3-47 lists and describes the fields in the Resource Manager Details page.

#### Table 3-47 HOME-AGENT Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HomeAgentIPAddresses tab</td>
<td></td>
</tr>
</tbody>
</table>
Click the **Add** button to save the details and list it in Start and End IP list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See **Relocating Records** for more details. To delete the available attributes, select the relevant attribute and click the **Delete** button below.

**HOME-AGENT-IPv6**

Table 3-48 lists and describes the fields in the Resource Manager Details page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>Required; must be an IPv6 address.</td>
</tr>
<tr>
<td>End</td>
<td>Required; must be an IPv6 address.</td>
</tr>
</tbody>
</table>

Click the **Add** button to save the details and list it in Start and End IPv6 list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See **Relocating Records** for more details. To delete the available attributes, select the relevant attribute and click the **Delete** button below.

**IP-DYNAMIC**

Table 3-49 lists and describes the fields in the Resource Manager Details page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General tab</td>
<td></td>
</tr>
<tr>
<td>Reuse IP for same SessionKey and User</td>
<td>When set to TRUE, this property supports overlapping IP addresses between session managers for VPN users. Default value is FALSE.</td>
</tr>
<tr>
<td>Net Mask</td>
<td>Required; must be set to a valid net mask.</td>
</tr>
<tr>
<td>Allow Overlapped IP Addresses</td>
<td>When set to TRUE, this property supports overlapping IP addresses between session managers for VPN users. Default value is FALSE.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IP Addresses tab</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>Required; must be an IP address.</td>
</tr>
<tr>
<td>End</td>
<td>Required; must be an IP address.</td>
</tr>
</tbody>
</table>

Click the **Add** button to save the details and list it in Start and End IP list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See **Relocating Records** for more details. To delete the available attributes, select the relevant attribute and click the **Delete** button below.

**REMOTE-IP-DYNAMIC**

Table 3-50 lists and describes the fields in the Resource Manager Details page.


Table 3-50  REMOTE-IP-DYNAMIC Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General tab</td>
<td></td>
</tr>
<tr>
<td>Reuse IP for same SessionKey and User</td>
<td>When set to TRUE, this property supports overlapping IP addresses between session managers for VPN users. Default value is FALSE.</td>
</tr>
<tr>
<td>Net Mask</td>
<td>Required; must be set to a valid net mask.</td>
</tr>
<tr>
<td>Allow Overlapped IP Addresses</td>
<td>When set to TRUE, this property supports overlapping IP addresses between session managers for VPN users. Default value is FALSE.</td>
</tr>
<tr>
<td>IP Addresses tab</td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>Required; must be an IP address.</td>
</tr>
<tr>
<td>End</td>
<td>Required; must be an IP address.</td>
</tr>
</tbody>
</table>

Click the **Add** button to save the details and list it in Start and End IP list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the **Delete** button below.

**IP-PER-NAS-PORT**

Table 3-51 lists and describes the fields in the Resource Manager Details page.

Table 3-51  IP-PER-NAS-PORT Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General tab</td>
<td></td>
</tr>
<tr>
<td>Net Mask</td>
<td>Required; if used, must be set to a valid net mask.</td>
</tr>
<tr>
<td>Allow Overlapped IP Addresses</td>
<td>When set to TRUE, this property supports overlapping IP addresses between session managers for VPN users. Default value is FALSE.</td>
</tr>
<tr>
<td>NAS</td>
<td>Required; must be the name of a known Client. This value must be the same as the NAS-Identifier attribute in the Access-Request packet.</td>
</tr>
<tr>
<td>IP Config tab</td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>Required; must be an IP address.</td>
</tr>
<tr>
<td>End</td>
<td>Required; must be an IP address.</td>
</tr>
<tr>
<td>Port Config tab</td>
<td></td>
</tr>
<tr>
<td>Start</td>
<td>Required; set the NAS port</td>
</tr>
<tr>
<td>End</td>
<td>Required; set the NAS port</td>
</tr>
</tbody>
</table>

Click the **Add** button to save the details and list it in Start and End IP list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the **Delete** button below.

**IPX-DYNAMIC**

Table 3-52 lists and describes the fields in the Resource Manager Details page.
Click the **Add** button to save the details and list it in Start and End IP list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See **Relocating Records** for more details. To delete the available attributes, select the relevant attribute and click the **Delete** button below.

### SESSION-CACHE

Table 3-53 lists and describes the fields in the Resource Manager Details page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General tab</strong></td>
<td></td>
</tr>
<tr>
<td>Overwrite Attributes</td>
<td>Specifies whether to overwrite the existing attributes if there are any in the session record.</td>
</tr>
<tr>
<td>Query Key</td>
<td>Required; set the QueryKey to the a RADIUS attribute you want to key on, such as Framed-IP-Address. A change made in Prime Access Registrar requires that this attribute not be an XML attribute, even if this session-cache resource manager is being used for an XML query.</td>
</tr>
<tr>
<td>Note</td>
<td>Any existing session-cache resource managers using an XML attribute for the Query Key must be changed to a RADIUS attribute that this XML attribute is mapped to under Query-Mappings.</td>
</tr>
<tr>
<td>Pending Removal Delay</td>
<td>Required; length of time information remains in the cache after the session ends (defaults to 10 seconds)</td>
</tr>
<tr>
<td><strong>Query Mapping tab</strong></td>
<td></td>
</tr>
<tr>
<td>XML Attribute</td>
<td>Set the QueryKey property to the XML attribute you want to key on such as XML-Address-format-IPv4 and list all attributes to be cached in the AttributesToBeCached subdirectory.</td>
</tr>
<tr>
<td>Radius Attribute</td>
<td>Required; list of attribute pairs, mapping the XML attributes on the left-hand side to the RADIUS attribute on the right-hand side.</td>
</tr>
<tr>
<td><strong>AttributeToBeCached tab</strong></td>
<td></td>
</tr>
<tr>
<td>RADIUS</td>
<td>Optional; set Radius, if the attribute needs to be defined for RADIUS.</td>
</tr>
<tr>
<td>VENDOR</td>
<td>Optional; set Vendor, if the attribute needs to be defined for Vendor. If Vendor is selected, specify the vendor type from the drop-down list.</td>
</tr>
<tr>
<td>Attribute Name</td>
<td>Required; use this subdirectory to provide a list of RADIUS attributes you want to store in cache</td>
</tr>
</tbody>
</table>
Click the Add button to save the details and list it in Start and End IP list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.

**SUBNET-DYNAMIC**

Table 3-54 lists and describes the fields in the Resource Manager Details page.

<table>
<thead>
<tr>
<th>Table 3-54  SUBNET-DYNAMIC Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fields</strong></td>
</tr>
<tr>
<td>Subnet Dynamic tab</td>
</tr>
<tr>
<td>Net Mask</td>
</tr>
<tr>
<td>Start</td>
</tr>
<tr>
<td>End</td>
</tr>
</tbody>
</table>

Click the Add button to save the details and list it in Start and End IP list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.

**USER-SESSION-LIMIT**

Table 3-55 lists and describes the fields in the Resource Manager Details page.

<table>
<thead>
<tr>
<th>Table 3-55  USER-SESSION-LIMIT Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fields</strong></td>
</tr>
<tr>
<td>User Session Limit</td>
</tr>
</tbody>
</table>

**REMOTE-USER-SESSION-LIMIT**

Table 3-56 lists and describes the fields in the Resource Manager Details page.

<table>
<thead>
<tr>
<th>Table 3-56  REMOTE-USER-SESSION-LIMIT Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fields</strong></td>
</tr>
<tr>
<td>User Session Limit</td>
</tr>
</tbody>
</table>

**USR-VPN**

Table 3-57 lists and describes the fields in the Resource Manager Details page.

<table>
<thead>
<tr>
<th>Table 3-57  USR-VPN Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fields</strong></td>
</tr>
<tr>
<td>General tab</td>
</tr>
<tr>
<td>Identifier</td>
</tr>
</tbody>
</table>
Configuring Cisco Prime Access Registrar

Chapter 3 Using the Graphical User Interface

To edit the gateway details, check the appropriate check box and click the **Edit** button. Enter new information in the editable fields and click the **Save** button. You can also delete the record using **Delete** button.

### Table 3-57 USR-VPN Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neighbor</td>
<td>Optional; if set, should be the IP address of the next hop router for the VPN.</td>
</tr>
<tr>
<td>Framed Routing</td>
<td>Optional; if set, should be <strong>RIP V2 Off</strong> or <strong>RIP V2 On</strong> if the USR NAS is to run RIP Version 2 for the user.</td>
</tr>
</tbody>
</table>

**Gateway tab**

<table>
<thead>
<tr>
<th>Name of Gateway</th>
<th>Required; name of the gateway.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description (optional)</td>
<td>Optional; description of the gateway.</td>
</tr>
<tr>
<td>IP Address</td>
<td>Required; IP address of the gateway</td>
</tr>
<tr>
<td>Shared Secret</td>
<td>Required; must match the shared secret of the gateway.</td>
</tr>
<tr>
<td>Tunnel Refresh</td>
<td>Optional; if specified it is the number of seconds the tunnel stays active before a secure “keepalive” is exchanged between the tunnel peers in order to maintain the tunnel open.</td>
</tr>
<tr>
<td>Location ID</td>
<td>Optional; if specified it is a string indicating the physical location of the gateway. Click the <strong>Save</strong> button, to save the details.</td>
</tr>
</tbody>
</table>

To edit the gateway details, check the appropriate check box and click the **Edit** button. Enter new information in the editable fields and click the **Save** button. You can also delete the record using **Delete** button.

### REMOTE-SESSION-CACHE

Table 3-58 lists and describes the fields in the Resource Manager Details page.

### Table 3-58 REMOTE-SESSION-CACHE Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General tab</td>
<td></td>
</tr>
<tr>
<td>Overwrite Attributes</td>
<td>Specifies whether to overwrite the existing attributes if there are any in the session record.</td>
</tr>
<tr>
<td>Query Key</td>
<td>Required; set the QueryKey to the a RADIUS attribute you want to key on, such as Framed-IP-Address.</td>
</tr>
<tr>
<td>Pending Removal Delay</td>
<td>Required; length of time information remains in the cache after the session ends (defaults to 10 seconds)</td>
</tr>
</tbody>
</table>

**Remote Query Mapping tab**

| XML Attribute         | Set the QueryKey property to the XML attribute you want to key on such as XML-Address-format-IPv4 and list all attributes to be cached in the AttributesToBeCached subdirectory. |
**Table 3-58** REMOTE-SESSION-CACHE Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radius Attribute</td>
<td>Required; list of attribute pairs, mapping the XML attributes on the left-hand side to the RADIUS attribute on the right-hand side.</td>
</tr>
</tbody>
</table>

**RemoteAttributeToBeCached tab**

| RADIUS                        | Optional; set Radius, if the attribute needs to be defined for RADIUS.      |
| VENDOR                        | Optional; set Vendor, if the attribute needs to be defined for Vendor. If Vendor is selected, specify the vendor type from the drop-down list. |
| Attribute Name                | Required; use this subdirectory to provide a list of RADIUS attributes you want to store in cache |

**3GPP**

Table 3-59 lists and describes the 3GPP properties in the Resource Manager Details page.

**Table 3-59** 3GPP Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnableRegistrationFlow</td>
<td>Check the box to enable initiation of a Server-Assignment-Request (SAR) registration message when a session is created and a SAR deregistration message when a session is deleted.</td>
</tr>
<tr>
<td>EnableSessionTermination</td>
<td>Check the box to enable initiation of a Server-Termination-Request (STR) message when a session is deleted.</td>
</tr>
<tr>
<td>ReuseExistingSession</td>
<td>If selected, SAR registration will not be initiated for an existing session.</td>
</tr>
<tr>
<td>HSSProxyService</td>
<td>Required; a service of type Diameter used to group a list of HSS/Diameter servers towards which the SAR and STR messages need to be initiated in the 3GPP authorization flow.</td>
</tr>
</tbody>
</table>

You can use the Resource Manager List page for the following:

- Filtering Records
- Adding Resource Manager Details
- Network Resources
- Editing Records
- Deleting Records

**Adding Resource Manager Details**

To add new resource manager details:

**Step 1** Choose **Configuration > Resource Manager**. The Resource Manager List page is displayed.

**Step 2** Click **Add**. The Resource Manager Details page is displayed.

**Step 3** Enter the required details.
Step 4  Click **Submit** to save the specified details in the Resource Manager Details page. Otherwise click **Cancel** to return to the Resource Manager List page without saving the details.

The Resource Manager List page is displayed with the newly added details or a respective error message is displayed.

---

**Note**

Resource Manager supports the following remote type session managers: remote-ip-dynamic, remote-session-cache, home-agent, remote-user-session-limit, home-agent-ipv6 and remote-group-session-limit.

---

## Network Resources

Network Resources constitutes the maintenance and management of the details of the clients and remote servers. The clients IP address and shared secret details are maintained under clients. The management of server directory with use of remote server protocols details are maintained in remote server.

This section describes the following:

- **Clients**
- **Remote Servers**

### Clients

All NASs and proxy clients that communicate directly with Prime Access Registrar must have an entry in the Clients list. This is required because NAS and proxy clients share a secret with the RADIUS server which is used to encrypt passwords and to sign responses.

Table 3-60 lists and describes the fields in the Client Details page.

**Table 3-60  Client Properties**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required and should match the Client identifier specified in the standard RADIUS attribute, <strong>NAS-Identifier</strong>. The name must be unique within the Clients list.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; you can use this property to specify a Script you can use to determine the services to use for authentication, authorization, and/or accounting.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; you can use this property to specify a Script you can use to make any Client-specific modifications when responding to a particular Client.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Required; set it to <strong>Radius</strong>, <strong>Diameter</strong>, or <strong>Tacacs-and-Radius</strong>.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the client.</td>
</tr>
<tr>
<td>Vendor</td>
<td>Optional; displayed when the protocol is set to Diameter. When set, must be the name of a known Vendor.</td>
</tr>
</tbody>
</table>
### Table 3-60 Client Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server Identity</td>
<td>Optional; displayed when the protocol is set to Diameter. While exchanging the CER information in the client, Prime Access Registrar sends the configured server identity value as the origin-host value. When set, it takes precedence over the /Realm/Advance/Diameter/TransportManagement configuration.</td>
</tr>
<tr>
<td>HostName</td>
<td>Required; hostname or IP address of the Diameter client.</td>
</tr>
<tr>
<td>Port</td>
<td>Required; port on which client connects with the Prime Access Registrar server.</td>
</tr>
<tr>
<td>SCTP-Enabled</td>
<td>Required; displays when the protocol is set to Diameter and indicates whether the connection will be an SCTP. If set to TRUE, SCTP will be used. If set to FALSE, TCP will be used.</td>
</tr>
<tr>
<td>Advertised-Realm</td>
<td>Optional; displays when the protocol is set to Diameter. While exchanging the CER information in the client, Prime Access Registrar sends the configured server realm value as the origin-realms value. It takes precedence over the /Realm/Advance/Diameter/TransportManagement configuration.</td>
</tr>
<tr>
<td>WatchDogTimeout</td>
<td>Time interval between watchdog messages.</td>
</tr>
<tr>
<td>MaxIncomingRequestRate</td>
<td>Maximum number of incoming requests allowed per second.</td>
</tr>
<tr>
<td>InitialTimeout</td>
<td>Timeout value, in milliseconds, the Prime Access Registrar server waits for a response before dropping the packet.</td>
</tr>
<tr>
<td>TLS-Enabled</td>
<td>Check this box to enable TLS security mechanism for the Diameter client.</td>
</tr>
</tbody>
</table>

#### TLSOptions

This section is available if the protocol is set to Diameter and TLS-Enabled option is checked.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PrivateKeyPassword</td>
<td>The password used to protect the server’s private key.</td>
</tr>
</tbody>
</table>
| ServerKeyFile           | The full pathname of the file containing the server’s RSA private key. The path name can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The valid encoding prefix is “PEM”. If an encoding prefix is not present, the file is assumed to be in PEM format. The following example assumes that the subdirectory pki under /cisco-ar contains the server’s certificate file. The file server-key.pem is assumed to be in PEM format. The file extension .pem is not significant.  
  
  ```shell
  set ServerKeyFile PEM:/cisco-ar/pki/server-key.pem
  ```  
| ServerCertificateFile   | The full pathname of the file containing the server’s certificate or certificate chain used during the TLS exchange. The path name can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The valid encoding prefix is PEM. If an encoding prefix is not present, the file is assumed to be in PEM format. |
| CACertificateFile       | The full pathname of the file containing trusted CA certificates used for client verification. The file can contain more than one certificate, but all certificates must be in PEM format. DER encoding is not allowed. |
CACertificatePath

The name of a directory containing trusted CA certificates (in PEM format) used for client verification. This parameter is optional, and if it is used there are some special preparations required for the directory it references.

Each certificate file in this directory must contain exactly one certificate in PEM format. The server looks up the certificate files using the MD5 hash value of the certificate’s subject name as a key. The directory must therefore also contain a set of symbolic links each of which points to an actual certificate file. The name of each symbolic link is the hash of the subject name of the certificate.

For example, if a certificate file named `ca-cert.pem` is located in the CACertificatePath directory, and the MD5 hash of the subject name contained in `ca-cert.path.pem` is 1b96dd93, then a symbolic link named 1b96dd93 must point to `ca-cert.pem`.

If there are subject name collisions such as multiple certificates with the same subject name, each link name must be indexed with a numeric extension as in 1b96dd93.0 and 1b96dd93.1.

<table>
<thead>
<tr>
<th>PeerVerificationMode</th>
<th>Select one of the following options:</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>If the peer verification certificate must not be requested.</td>
</tr>
<tr>
<td>Optional</td>
<td>If peer verification certificate can be requested; but, verification is not required.</td>
</tr>
<tr>
<td>RequireCertificate</td>
<td>If peer certificate must be requested and verified.</td>
</tr>
</tbody>
</table>

VerificationDepth

Specifies the maximum length of the certificate chain used for client verification.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CACertificatePath</td>
<td>The name of a directory containing trusted CA certificates (in PEM format) used for client verification. This parameter is optional, and if it is used there are some special preparations required for the directory it references. Each certificate file in this directory must contain exactly one certificate in PEM format. The server looks up the certificate files using the MD5 hash value of the certificate’s subject name as a key. The directory must therefore also contain a set of symbolic links each of which points to an actual certificate file. The name of each symbolic link is the hash of the subject name of the certificate. For example, if a certificate file named <code>ca-cert.pem</code> is located in the CACertificatePath directory, and the MD5 hash of the subject name contained in <code>ca-cert.path.pem</code> is 1b96dd93, then a symbolic link named 1b96dd93 must point to <code>ca-cert.pem</code>. If there are subject name collisions such as multiple certificates with the same subject name, each link name must be indexed with a numeric extension as in 1b96dd93.0 and 1b96dd93.1.</td>
</tr>
<tr>
<td>PeerVerificationMode</td>
<td>Select one of the following options:</td>
</tr>
<tr>
<td>VerificationDepth</td>
<td>Specifies the maximum length of the certificate chain used for client verification.</td>
</tr>
</tbody>
</table>
### Table 3-60  
**Client Properties (continued)**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnableAuto-Chaining</td>
<td>When set to TRUE, Prime Access Registrar sends its server certificate chain (Server-Cert -&gt; IntermediateCA -&gt; RootCA) while presenting the server certificate to the client for server side authentication. When set to FALSE, Prime Access Registrar sends only the server certificate (Server-Cert) to the client.</td>
</tr>
</tbody>
</table>

**General Properties tab**  
The tabs are available if the protocol is set to Radius or Tacacs-and-Radius.

| IPAddress            | Required; must be a valid IP address and unique in the Clients list. Prime Access Registrar uses this property to identify the Client that sent the request, either using the source IP address to identify the immediate sender or using the **NAS-IP-Address** attribute in the Request dictionary to identify the NAS sending the request through a proxy.  
> When a range is configured for a Client’s IPAddress property, any incoming requests whose source address belongs to the range specified, will be allowed for further processing by the server. Similarly when a wildcard (an asterisk ‘*’ in this case) is specified, any incoming requests whose source address matches the wildcard specification will be allowed. In both the cases, the configured client properties like SharedSecret, and Vendor are used to process the requests.  
> You can specify a range of IP addresses using a hyphen as in:  
> 100.1.2.11-20  
> You can use an asterisk wildcard to match all numbers in an IP address octet as in:  
> 100.1.2.*  
> You can specify an IPAddress and a subnet mask together using Classless Inter-Domain Routing (CIDR) notation as in:  
> 100.1.2.0/24  
> You can use the IPAddress property to set a base address and use the NetMask property to specify the number of clients in the subnet range. |
| Shared Secret        | Required; must match the secret configured in the Client.                                                                                                                                                  |
| Type                 | Required; accept the default (NAS), or set it to ATM, Proxy, or NAS+Proxy.                                                                                                                                |
| Vendor               | Optional; you can use this property when you need special processing for a specific vendor’s NAS. To use this property, you must configure a **Vendor** object and include a script. Prime Access Registrar provides five Scripts you can use: one for Ascend, Cisco, Cabletron, Altiga, and one for USR. You can also provide your own Script. |
Network Resources

Chapter 3      Using the Graphical User Interface

Table 3-60  Client Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
</table>
| NetMask             | Specifies the subnet mask used with the network address setting configured for the IPAddress property when configuring a range of IP addresses. This property is not used for a single client with an IP address only. The NetMask property is used to configure multiple clients when you configure a base IP address in the IPAddress property. You can set the NetMask property for a range of 256 clients using the following example: set NetMask 255.255.255.0  
  
  Note  If you set the NetMask property, validation will fail if you attempt to specify a subnet mask using CIDR notation with the IPAddress property (described above). |
| Enforce Traffic Throttling | By default, the value is set to FALSE. When set to TRUE, the traffic throttling check for the packet will be executed. |

Dynamic Authorization tab

| Enable Dynamic Authorization | Optional; when set to TRUE, this property enables Change of Authorization (CoA) and Packet of Disconnect (PoD) features. |
| Shared Secret              | Located under the DynamicAuthorizationServer subdirectory, this is the shared secret used for communicating CoA and PoD packets with the client. |
| Port                        | Located under the DynamicAuthorizationServer subdirectory, the default port is 3799. |
| InitialTimeout              | Located under the DynamicAuthorizationServer subdirectory, the default is 5000. |
| MaxTries                    | Located under the DynamicAuthorizationServer subdirectory, the default is 3. |
| COA Attribute               | This property is found under the DynamicAuthorizationServer subdirectory and points to a group of attributes to be included in a CoA request sent to this client. These attribute groups are created and configured under the AttributeGroups subdirectory in /Radius/Advanced. |
| POD Attribute               | This property is found under the DynamicAuthorizationServer subdirectory and points to a group of attributes to be included in a POD request sent to this client. These attribute groups are created and configured under the AttributeGroups subdirectory in /Radius/Advanced. |

Notification Properties tab

| Enable Notifications | Required; the default value is FALSE and indicates the client is not capable of receiving Accounting-Stop notifications from the Prime Access Registrar server. When set to TRUE, the client can receive Accounting-Stop notifications from the Prime Access Registrar server and additional properties must be configured under a new sub-directory named NotificationProperties. |
| InitialTimeout        | Located under the NotificationProperties subdirectory, specifies the timeout value in milliseconds the Prime Access Registrar server waits for an Accounting-Response packet before attempting a retry (sending another Accounting-Stop packet to the client). Required when EnableNotifications is set to TRUE; the default value is 5000. |
You can use the Clients page for the following:

- Filtering Records
- Adding Client Details
- Editing Records
- Deleting Records

### Adding Client Details

To add new Client details:

**Step 1** Choose **Network Resources > Clients**. The Clients page is displayed.

**Step 2** Click **Add** to add new client details. The Client Details page is displayed.

**Step 3** Enter the required details in the General Properties, Dynamic Authorization, and Notification Properties tabs.

**Step 4** Click **Save** to save the specified details in the Client Details page. Otherwise click **Cancel** to return to the Client page without saving the details.

The Client page is displayed with the newly added details or a respective error message is displayed.

### Remote Servers

You can use the RemoteServers object to specify the properties of the remote servers to which Services proxy requests.
Prime Access Registrar provides the following RemoteServer protocol types:

- LDAP
- LDAP Accounting
- ODBC/OCI
- ODBC/OCI-Accounting
- Diameter
- Others

**Note**

You must not configure a remote server with an IP address, which is same as that of the client. This is applicable for all types of remote servers.

### LDAP

Specify the `ldap` service type when you want to use a particular LDAP remote server for authentication and/or authorization. When using LDAP for authentication and a local database for authorization, ensure that the usernames in both locations are identical with regard to case-sensitivity.

Table 3-61 lists and describes the fields in the Add LDAP-RemoteServers Details page.

**Table 3-61**  **LDAP Server Properties**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LDAP Properties tab</strong></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Required; name of the LDAP server.</td>
</tr>
<tr>
<td>Host Name</td>
<td>Required; the LDAP server’s hostname or IP address. Prime Access Registrar supports IPv4 and IPv6 addresses for the hostname.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> To use IPv6 addresses, you must have Next Generation (NG) license of Prime Access Registrar. For LDAP, IPv6 addresses must be enclosed in square brackets, as in [2001:420:27c1:420:250:56ff:fe99:3dfd].</td>
</tr>
<tr>
<td>Port</td>
<td>Required; defaults to port 389.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the LDAP server.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Required; the default is 15. The timeout property indicates how many seconds the RADIUS server will wait for a response from the LDAP server. <strong>Note</strong> Use InitialTimeout from above as a template, except this is timeout is specified in seconds.</td>
</tr>
<tr>
<td>Reactivate Time Interval</td>
<td>Required; the amount of time (in milliseconds) to wait before retrying a remote server that was offline. You must specify a number greater than zero. The default is 300,000 (5 minutes).</td>
</tr>
</tbody>
</table>
MaxReferrals | Required; must be a number equal to or greater than zero. This property indicates how many referrals are allowed when looking up user information. When you set this property to zero, no referrals are allowed. Cisco Prime Access Registrar manages referrals by allowing the RADIUS server’s administrator to indicate an LDAP “referral attribute,” which might or might not appear in the user information returned from an LDAP query. When this information is returned from a query, Cisco Prime Access Registrar assumes it is a referral and initiates another query based on the referral. Referrals can also contain referrals.  
**Note** This is an LDAP v2 referral property.

Referral Attribute | Required when you have specified a MaxReferrals value. This property specifies which LDAP attribute, returned from an LDAP search, to check for referral information.  
**Note** This is an LDAP v2 referral property.

Referral Filter | Required when you have specified a MaxReferrals value. This is the filter Cisco Prime Access Registrar uses when processing referrals. When checking referrals, the information Cisco Prime Access Registrar finds in the referral itself is considered to be the search path and this property provides the filter. The syntax is the same as that of the Filter property.  
**Note** This is an LDAP v2 referral property.

Bind Name | Optional; the distinguished name (dn) to use when establishing a connection between the LDAP and RADIUS servers.

Bind Password | Optional; the password associated with the BindName.

Search Path | Required; the path that indicates where in the LDAP database to start the search for user information.

Limit Outstanding Requests | Required; the default is FALSE. Cisco Prime Access Registrar uses this property in conjunction with the MaxOutstandingRequests property to tune the RADIUS server’s use of the LDAP server. When you set this property to TRUE, the number of outstanding requests for this RemoteServer is limited to the value you specified in MaxOutstandingRequests. When the number of requests exceeds this number, Cisco Prime Access Registrar queues the remaining requests, and sends them as soon as the number of outstanding requests drops to this number.

User Password Attribute | Required; this specifies which LDAP field the RADIUS server should check for the user’s password.

Escape Spl.Character in UserName | FALSE by default

Datasource Connections | Specifies the number of concurrent connections to the LDAP server. The default value is 8.

**Table 3-61 LDAP Server Properties (continued)**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
</table>
| MaxReferrals                  | Required; must be a number equal to or greater than zero. This property indicates how many referrals are allowed when looking up user information. When you set this property to zero, no referrals are allowed. Cisco Prime Access Registrar manages referrals by allowing the RADIUS server’s administrator to indicate an LDAP “referral attribute,” which might or might not appear in the user information returned from an LDAP query. When this information is returned from a query, Cisco Prime Access Registrar assumes it is a referral and initiates another query based on the referral. Referrals can also contain referrals.  
**Note** This is an LDAP v2 referral property. |
| Referral Attribute            | Required when you have specified a MaxReferrals value. This property specifies which LDAP attribute, returned from an LDAP search, to check for referral information.  
**Note** This is an LDAP v2 referral property. |
| Referral Filter               | Required when you have specified a MaxReferrals value. This is the filter Cisco Prime Access Registrar uses when processing referrals. When checking referrals, the information Cisco Prime Access Registrar finds in the referral itself is considered to be the search path and this property provides the filter. The syntax is the same as that of the Filter property.  
**Note** This is an LDAP v2 referral property. |
<p>| Bind Name                     | Optional; the distinguished name (dn) to use when establishing a connection between the LDAP and RADIUS servers. |
| Bind Password                 | Optional; the password associated with the BindName. |
| Search Path                   | Required; the path that indicates where in the LDAP database to start the search for user information. |
| Limit Outstanding Requests    | Required; the default is FALSE. Cisco Prime Access Registrar uses this property in conjunction with the MaxOutstandingRequests property to tune the RADIUS server’s use of the LDAP server. When you set this property to TRUE, the number of outstanding requests for this RemoteServer is limited to the value you specified in MaxOutstandingRequests. When the number of requests exceeds this number, Cisco Prime Access Registrar queues the remaining requests, and sends them as soon as the number of outstanding requests drops to this number. |
| User Password Attribute       | Required; this specifies which LDAP field the RADIUS server should check for the user’s password. |
| Escape Spl.Character in UserName | FALSE by default |
| Datasource Connections        | Specifies the number of concurrent connections to the LDAP server. The default value is 8. |</p>
<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use SSL</td>
<td>A boolean field indicating whether you want Cisco Prime Access Registrar to use SSL (Secure Socket Layer) when communicating with this RemoteServer. When you set it to TRUE, be sure to specify the CertificateDBPath field in the Advanced section, and be sure the port you specified for this RemoteServer is the SSL port used by the LDAP server.</td>
</tr>
<tr>
<td>EnableKeepAlive</td>
<td>Default is FALSE. This is enabled to send a TCP keepalive to keep the idle connection active.</td>
</tr>
<tr>
<td>Filter</td>
<td>Required; this specifies the search filter Cisco Prime Access Registrar uses when querying the LDAP server for user information. When you configure this property, use the notation “%s” to indicate where the user ID should be inserted. For example, a typical value for this property is “(uid=%s),” which means that when querying for information about user joe, use the filter uid=joe.</td>
</tr>
<tr>
<td>Max Outstanding Requests</td>
<td>Required when you have set the LimitOutstandingRequests to TRUE. The number you specify, which must be greater than zero, determines the maximum number of outstanding requests allowed for this remote server.</td>
</tr>
<tr>
<td>Password Encryption Style</td>
<td>The default is None. You can also specify crypt, dynamic, SHA-1, and SSHA-1.</td>
</tr>
<tr>
<td>DNSLookup and LDAP RebindInterval</td>
<td>Specifies the timeout period after which the Prime Access Registrar server will attempt to resolve the LDAP hostname to IP address (DNS resolution); 0 by default</td>
</tr>
<tr>
<td>Search Scope</td>
<td>Specifies how deep to search within a search path; default is SubTree which indicates a search of the base object and the entire subtree of which the base object distinguished name is the highest object. \Base indicates a search of the base object only. \OneLevel indicates a search of objects immediately subordinate to the base object, but does not include the base object.</td>
</tr>
<tr>
<td>Use Binary Password Comparison</td>
<td>A boolean field that enables binary password comparison for authentication. This property when set to TRUE, enables binary password comparison. By default, this property is set to FALSE.</td>
</tr>
<tr>
<td>Use Bind Based Authentication</td>
<td>A boolean field that enables bind-based authentication with LDAP server. By default, this property is set to FALSE. When set to FALSE, it uses existing legacy authentication method. On setting this property to TRUE, the mappings LDAPToRadius, LDAPToEnvironment, and LDAPToCheckItem will not work.</td>
</tr>
<tr>
<td>LDAPToRadiusMappings tab</td>
<td></td>
</tr>
<tr>
<td>LDAPAttribute</td>
<td>Set the value for the LDAP attribute</td>
</tr>
</tbody>
</table>
Table 3-61  LDAP Server Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RadiusAttribute</td>
<td>A list of name/value pairs in which the name is the name of the ldap attribute to retrieve from the user record, and the value is the name of the RADIUS attribute to set to the value of the ldap attribute retrieved.</td>
</tr>
<tr>
<td></td>
<td>For example, when the LDAPToRadiusMappings has the entry: FramedIPAddress = Framed-IP-Address, the RemoteServer retrieves the FramedIPAddress attribute from the ldap user entry for the specified user, uses the value returned, and sets the Response variable Framed-IP-Address to that value.</td>
</tr>
<tr>
<td></td>
<td>Click the Add button to save the details and list it in the attribute list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
</tbody>
</table>

LDAPToCheckItems Mappings tab

<table>
<thead>
<tr>
<th>Attribute Type</th>
<th>Select either RADIUS or VENDOR. If Vendor is selected, specify the vendor type from the drop-down list.</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDAPAttribute</td>
<td>Set the value for the LDAP attribute</td>
</tr>
<tr>
<td>CheckedItems</td>
<td>A list of LDAP attribute/value pairs which must be present in the RADIUS access request and must match, both name and value, for the check to pass.</td>
</tr>
<tr>
<td></td>
<td>For example, when the LDAPToCheckItemMappings has the entry: group = User-Group, the Access Request must contain the attribute group, and it must be set to User-Group.</td>
</tr>
<tr>
<td></td>
<td>Click the Add button to save the details and list it in the attribute list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
</tbody>
</table>

LDAPToEnvironmentalMappings tab

<table>
<thead>
<tr>
<th>LDAPAttribute</th>
<th>Set the value for the LDAP attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnvironmentalAttribute</td>
<td>A list of name/value pairs in which the name is the name of the ldap attribute to retrieve from the user record, and the value is the name of the Environment variable to set to the value of the ldap attribute retrieved.</td>
</tr>
<tr>
<td></td>
<td>For example, when the LDAPToEnvironmentMappings has the entry: group = User-Group, the RemoteServer retrieves the group attribute from the ldap user entry for the specified user, uses the value returned, and sets the Environment variable User-Group to that value.</td>
</tr>
<tr>
<td></td>
<td>Click the Add button to save the details and list it in the attribute list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
</tbody>
</table>
You can use the LDAP-RemoteServers page for the following:

- Filtering Records
- Adding LDAP Details
- LDAP Accounting
- Editing Records
- Deleting Records

Adding LDAP Details

To add new LDAP details:

**Step 1** Choose **Network Resources > RemoteServers > LDAP**. The LDAP-RemoteServers page is displayed.

**Step 2** Click **Add** to add LDAP details. The LDAP-RemoteServers Details page is displayed.

**Step 3** Enter the required details in the tabs.

**Step 4** Click **Save LDAP Server** to save the specified details in the LDAP-RemoteServers Details page. The LDAP-RemoteServers page is displayed with the newly added details or a respective error message is displayed. Otherwise click **Cancel** to return to the LDAP-RemoteServers page without saving the details.

LDAP Accounting

Previous releases of Prime Access Registrar supported accessing user data from an LDAP server, but this feature was limited to performing authentication and authorization (AA). You could only write the accounting records to local file or oracle database or proxy to another RADIUS server. Prime Access Registrar supports writing accounting records into LDAP server enabling integration between billing systems and LDAP.

Table 3-62 lists and describes the fields in the LDAPAcct RemoteServer Details page.

<table>
<thead>
<tr>
<th><strong>Table 3-62</strong> LDAP Accounting Server Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fields</strong></td>
</tr>
<tr>
<td><strong>LDAP Acct Properties tab</strong></td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>HostName</td>
</tr>
<tr>
<td>Port</td>
</tr>
<tr>
<td>Timeout</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
</tr>
</tbody>
</table>
### Table 3-62  LDAP Accounting Server Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BindName</td>
<td>Optional; the distinguished name (dn) to use when establishing a connection between the LDAP and RADIUS servers.</td>
</tr>
<tr>
<td>BindPassword</td>
<td>Optional; the password associated with the BindName.</td>
</tr>
<tr>
<td>EnableKeepAlive</td>
<td>Required; default is FALSE. This is enabled to send a TCP keepalive to keep the idle connection active.</td>
</tr>
<tr>
<td>Delimiter</td>
<td>Character used to separate the values of the attributes given in Attribute-List property.</td>
</tr>
<tr>
<td>LDAPEnvironmentMultiValueDelim</td>
<td>Optional; allows you to specify a character that separates multi-valued attribute lists when using ldap-accounting.</td>
</tr>
<tr>
<td>DnPath</td>
<td>Required; the path that indicates where in the LDAP database to start the write for user information.</td>
</tr>
<tr>
<td>EntryName</td>
<td>Required; this specifies the write entry name Prime Access Registrar uses when inserting the LDAP server for user information. When you configure this property, use the notation &quot;%(s)&quot; to indicate where the user ID should be inserted. For example, a typical value for this property is &quot;(uid=%s),&quot; which means that when inserting for information about user joe, use the entry name uid=joe.</td>
</tr>
<tr>
<td>LimitOutstandingRequests</td>
<td>Required; the default is FALSE. Prime Access Registrar uses this property in conjunction with the MaxOutstandingRequests property to tune the RADIUS server's use of the LDAP server. When you set this property to TRUE, the number of outstanding requests for this RemoteServer is limited to the value you specified in MaxOutstandingRequests. When the number of requests exceeds this number, Prime Access Registrar queues the remaining requests, and sends them as soon as the number of outstanding requests drops to this number.</td>
</tr>
<tr>
<td>MaxOutstandingRequests</td>
<td>Required when you have set the LimitOutstandingRequests to TRUE. The number you specify, which must be greater than zero, determines the maximum number of outstanding requests allowed for this remote server.</td>
</tr>
<tr>
<td>ObjectClass</td>
<td>Required; list of object classes which are all schemas defined in LDAP server. These schemas define required attributes and allowed attributes for an entry which is inserted from Prime Access Registrar.</td>
</tr>
<tr>
<td>DNSLookup and LDAPAcct RebindInterval</td>
<td>Specifies the timeout period after which the Prime Access Registrar server will attempt to resolve the LDAP hostname to IP address (DNS resolution).</td>
</tr>
<tr>
<td>Escape Spl.Character in UserName</td>
<td>FALSE by default.</td>
</tr>
<tr>
<td>AttributeList</td>
<td>List of comma-separated attribute names.</td>
</tr>
<tr>
<td>Datasource Connections</td>
<td>Mandatory number of connections to be established; defaults to 8.</td>
</tr>
<tr>
<td>UseLocalTimeZone</td>
<td>Optional; the default is FALSE. It determines the timezone of accounting records TimeStamp.</td>
</tr>
</tbody>
</table>
You can use the LDAP Acct-RemoteServers page for the following:

- Filtering Records
- Adding LDAP Accounting Details
- Editing Records
- Deleting Records

**Adding LDAP Accounting Details**

To add new LDAP accounting details:

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Choose <strong>Network Resources &gt; RemoteServers &gt; LDAP Accounting</strong>. The LDAPAcct-RemoteServers page is displayed.</td>
</tr>
<tr>
<td>Step 2</td>
<td>Click <strong>Add</strong> to add LDAP accounting details. The LDAPAcct RemoteServer Details page is displayed.</td>
</tr>
<tr>
<td>Step 3</td>
<td>Enter the required details in the tabs.</td>
</tr>
<tr>
<td>Step 4</td>
<td>Click <strong>Save LDAP Acct Server</strong> to save the specified details in the LDAPAcct RemoteServer Details page. Otherwise click <strong>Cancel</strong> to return to the LDAPAcct-RemoteServers page without saving the details.</td>
</tr>
</tbody>
</table>

The LDAPAcct-RemoteServers page is displayed with the newly added details or a respective error message is displayed.
ODBC/OCI

Specify `odbc` or `oci` when you want to use an ODBC or OCI service for authentication, authorization and accounting through an ODBC or OCI data store respectively. Use an ODBC or OCI service to authenticate and authorize an access requests by querying user information through ODBC or OCI and to insert accounting records into a data store through ODBC or OCI.

The ODBC service supports MYSQL and Oracle database service and OCI supports Oracle with 10.2.0 to 11.2.0 Oracle client.

Table 3-63 lists and describes the fields in the ODBC/OCI-RemoteServers Details page.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; name of the ODBC/OCI Server.</td>
</tr>
<tr>
<td>Protocol</td>
<td>The type of remote server. You select the option ODBC or OCI from the drop-down list.</td>
</tr>
<tr>
<td>Datasource Connections</td>
<td>Required; default is 8. This represents the total number of connections Prime Access Registrar can open with the ODBC server; total number of threads Prime Access Registrar can create for the ODBC server.</td>
</tr>
<tr>
<td>ODBC Datasource Name</td>
<td>Required; name of the ODBCDataSource to use and must refer to one entry in the list of ODBC datasources configured under <code>/Radius/Advanced/ODBCDataSources</code>.</td>
</tr>
<tr>
<td>User Password Attribute</td>
<td>Set the user password.</td>
</tr>
<tr>
<td>SNMPTrapIP</td>
<td>The SNMP trap IP for the remote servers.</td>
</tr>
<tr>
<td></td>
<td>Prime Access Registrar supports IPv4 and IPv6 addresses for the SNMP trap IP.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> To use IPv6 addresses, you must have Next Generation (NG) license of Prime Access Registrar.</td>
</tr>
<tr>
<td>Description</td>
<td>Description of the ODBC Server.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Required; the default is 15. The timeout property indicates how many seconds the RADIUS server will wait for a response from the ODBC server.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> Use InitialTimeout from above as a template, except this is timeout is specified in seconds.</td>
</tr>
<tr>
<td>Reactivate Time Interval</td>
<td>Required; default is 300,000 milliseconds. Length of time to wait before attempting to reconnect if a thread is not connected to a data source.</td>
</tr>
<tr>
<td>Keep Alive Timer Interval</td>
<td>Mandatory time interval (in milliseconds) to send a keepalive to keep the idle connection active; defaults to zero (0) meaning the option is disabled</td>
</tr>
<tr>
<td>SNMPTrapPort</td>
<td>The SNMP trap port for the remote server; defaults to 1521.</td>
</tr>
</tbody>
</table>

**SQL Definitions tab**

**Name**

SQLDefinition properties define the SQL you want to execute.
Network Resources

Table 3-63  ODBC/OCI Server Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Description of the SQL</td>
</tr>
<tr>
<td>Type</td>
<td>Prime Access Registrar supports only type query.</td>
</tr>
<tr>
<td>SQL</td>
<td>SQL query used to add, update or delete a record from a database</td>
</tr>
<tr>
<td>Execution SequenceNumber</td>
<td>Sequence number for SQL.Statement execution, must be greater than zero (mandatory, no default)</td>
</tr>
<tr>
<td>Marker List</td>
<td>Defines all markers for the query. MarkerList uses the format UserName/SQL_DATA_TYPE.</td>
</tr>
</tbody>
</table>

**RadiusMappings tab**

<table>
<thead>
<tr>
<th>ODBC/OCI Attribute</th>
<th>Set the ODBC or OCI attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>RADIUS Attribute</td>
<td>A list of name and value pairs in which the name is the name of the data store attribute to retrieve from the user record, and the value is the name of the RADIUS attribute to set to the value of the data store attribute retrieved. The data store attributes must match those defined in the external SQL file. Click the Add button to save the details and list it in the Attributes list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
</tbody>
</table>

**CheckItemsMappings tab**

<table>
<thead>
<tr>
<th>Attribute Type</th>
<th>Select either RADIUS or VENDOR. If Vendor is selected, specify the vendor type from the drop-down list.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ODBC/OCI Attribute</td>
<td>Set the ODBC or OCI attribute</td>
</tr>
<tr>
<td>CheckItem</td>
<td>A list of ODBC attribute/value pairs. Click the Add button to save the details and list it in the Attributes list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
</tbody>
</table>

**EnvironmentalMappings tab**

<table>
<thead>
<tr>
<th>ODBC/OCI Attribute</th>
<th>Set the ODBC or OCI attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Environmental Attribute</td>
<td>A list of name/value pairs in which the name is the name of the data store attribute to retrieve from the user record, and the value is the name of the Environment variable to set to the value of the ODBC attribute retrieved. Click the Add button to save the details and list it in the Attributes list. To navigate between the listed attributes, use the navigation option available adjacent to the list. See Relocating Records for more details. To delete the available attributes, select the relevant attribute and click the Delete button below.</td>
</tr>
</tbody>
</table>

You can use the ODBC/OCI-RemoteServers page for the following:
Network Resources

- Filtering Records
- Adding ODBC/OCI Details
- ODBC/OCI-Accounting
- Editing Records
- Deleting Records

### Adding ODBC/OCI Details

To add new ODBC or OCI details:

**Step 1** Choose **Network Resources > RemoteServers > ODBC/OCI**. The ODBC/OCI-RemoteServers page is displayed.

**Step 2** Click **Add** to add ODBC or OCI details. The ODBC/OCI-RemoteServers Details page is displayed.

**Step 3** Enter the required details.

**Step 4** Click **Add** to enter the SQL details in the **SQL Definitions** tab.

**Step 5** Click **Save** to save the specified details in the **SQL Definitions** tab or click **Cancel** to cancel the action.

**Step 6** Enter the required details in the tabs.

**Step 7** Click **Add Server** to save the specified details in the ODBC/OCI-RemoteServers Details page. Otherwise click **Cancel** to return to the ODBC/OCI-RemoteServers page without saving the details.

The ODBC/OCI-RemoteServers page is displayed with the newly added details or a respective error message is displayed.

### ODBC/OCI-Accounting

If you use the Oracle Accounting feature, you must configure an ODBC/OCI-Accounting RemoteServer object.

Table 3-64 lists and describes the fields in the Add ODBC/OCI Accounting-RemoteServers page.

**Table 3-64 ODBC/OCI Accounting Server Properties**

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General Properties tab</strong></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Name of the remote server; this property is mandatory, and there is no default.</td>
</tr>
<tr>
<td>Protocol</td>
<td>The type of Accounting remote server. You can select the option odbc-accounting or oci-accounting from the drop-down list.</td>
</tr>
<tr>
<td>Datasource Connections</td>
<td>Mandatory number of connections to be established; defaults to 8</td>
</tr>
<tr>
<td>ODBC Datasource Name</td>
<td>Name of the ODBCDataSource to use and must refer to one entry in the list of ODBC datasources configured under /Radius/Advanced/ODBCDataSources. Mandatory; no default</td>
</tr>
</tbody>
</table>
### Network Resources

**Chapter 3  Using the Graphical User Interface**

#### Table 3-64  ODBC/OCI Accounting Server Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buffer Accounting Packets</td>
<td>Mandatory, TRUE or FALSE, determines whether to buffer the accounting packets to local file, defaults to TRUE which means that packet buffering is enabled.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> When set to TRUE, a constant flow of incoming accounting packets can fill the buffer backing store files in /Cisco_AR/data/odbc beyond the size configured in MaximumBufferSize. Configure BackingStoreDiscThreshold in /Radius/Advanced when using ODBC accounting.</td>
</tr>
<tr>
<td>Max. Buffer Filesize</td>
<td>Mandatory if BufferAccountingPackets is set to TRUE, determines the maximum buffer file size, defaults to 10 Megabyte)</td>
</tr>
<tr>
<td>Backing Store Environment Variables</td>
<td>Optional; when BufferAccountingPackets is set to TRUE, contains a comma-separated list of environment variable names to be stored into a local file along with buffered packet. No default. BackingStoreEnvironmentVariables can also be specified in scripts using the BackingStoreEnvironmentVariables environment variable.</td>
</tr>
<tr>
<td>Attribute List</td>
<td>List of comma-separated attribute names.</td>
</tr>
<tr>
<td>SNMPTrapIP</td>
<td>Optional; when set to a valid IP address, the traps (responding/not responding traps) for the ODBC/OCI Accounting server will have this IP address. This is used to identify the server. If the value is not set, SNMP traps use 255.255.255.255 as the IP address.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of server.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Mandatory time interval (in seconds) to wait for SQL operation to complete; defaults to 15 seconds.</td>
</tr>
<tr>
<td>Reactivate Time Interval</td>
<td>Mandatory time interval (in milliseconds) to activate an inactive server; defaults to 300000 ms.</td>
</tr>
<tr>
<td>Keep Alive Timer Interval</td>
<td>Mandatory time interval (in milliseconds) to send a keepalive to keep the idle connection active; defaults to zero (0) meaning the option is disabled.</td>
</tr>
<tr>
<td>No. of Retries for Buffered Packet</td>
<td>Mandatory if BufferAccountingPackets is set to TRUE. A number greater than zero determines the number of attempts to be made to insert the buffered packet into Oracle. Defaults to 3.</td>
</tr>
<tr>
<td>Use Local Timezone</td>
<td>Set to TRUE or FALSE, determines the timezone of accounting records' TimeStamp (defaults to FALSE).</td>
</tr>
<tr>
<td>Delimiter</td>
<td>Character used to separate the values of the attributes given in AttributeList property.</td>
</tr>
<tr>
<td>SNMPTrapPort</td>
<td>Optional; when set to a valid port, the traps (responding/not responding traps) for the ODBC/OCI Accounting server will have this port. If the value is not set, SNMP traps use 1521 as the IP port.</td>
</tr>
</tbody>
</table>

**SQL Definitions tab**

<table>
<thead>
<tr>
<th>Name</th>
<th>Required; SQLDefinition properties define the SQL you want to execute.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Description of the SQL</td>
</tr>
</tbody>
</table>
Chapter 3  Using the Graphical User Interface

Network Resources

You can use the ODBC/OCI Accounting-RemoteServers page for the following:

- Filtering Records
- Adding ODBC/OCI Accounting Details
- Others
- Editing Records
- Deleting Records

### Adding ODBC/OCI Accounting Details

To add new ODBC or OCI accounting details:

**Step 1** Choose Network Resources > RemoteServers > ODBC/OCI Accounting. The ODBC/OCI Accounting-RemoteServers page is displayed.

**Step 2** Click Add to add ODBC or OCI accounting details. The ODBC/OCI Accounting-RemoteServers Details page is displayed.

**Step 3** Enter the required details in the tabs.

**Step 4** Click Add Accounting Server to save the specified details in the ODBC/OCI Accounting-RemoteServers Details page. The ODBC/OCI Accounting-RemoteServers page is displayed with the newly added details or a respective error message is displayed. Otherwise click Cancel to return to the ODBC/OCI Accounting-RemoteServers page without saving the details.

### Diameter

Diameter is a networking protocol which is derived from RADIUS protocol.

You can click the Add button in the Diameter-RemoteServers page to add a new Diameter remote server. Table 3-65 lists and describes the Diameter remote server properties.

---

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required; Prime Access Registrar supports insert, update and delete options.</td>
</tr>
<tr>
<td>SQL</td>
<td>Required; SQL query used to acquire the password</td>
</tr>
<tr>
<td>Execution SequenceNumber</td>
<td>Required; sequence number for SQLStatement execution, must be greater than zero (mandatory, no default)</td>
</tr>
<tr>
<td>Marker List</td>
<td>Required; defines all markers for the query. MarkerList uses the format UserName/SQL_DATA_TYPE.</td>
</tr>
</tbody>
</table>
### Table 3-65  Diameter Remote Server Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; name of the Diameter server.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the Diameter server.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Required; protocol used by the Diameter server.</td>
</tr>
<tr>
<td>MaxTries</td>
<td>Number of retry attempts to be made by the Diameter server for request and response.</td>
</tr>
<tr>
<td>Host Name</td>
<td>Host name of the server.</td>
</tr>
<tr>
<td>Initial Timeout</td>
<td>Specifies the timeout value in milliseconds the Prime Access Registrar server waits for an Accounting-Response packet before attempting a retry. This value must be less than the DWatchDogTimeout value.</td>
</tr>
<tr>
<td>Port</td>
<td>Port used by the server.</td>
</tr>
<tr>
<td>DWatchDogTimeout</td>
<td>Time interval between watch dog messages.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; if there is a script, it is the first script Prime Access Registrar runs when it receives a request from any client and/or for any service.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; if there is a script, it is the last script Prime Access Registrar runs before it sends a response to any client.</td>
</tr>
<tr>
<td>SCTP-Enabled</td>
<td>Indicates whether the connection will be an SCTP. If set to TRUE, SCTP will be used. If set to FALSE, TCP will be used.</td>
</tr>
<tr>
<td>AdvertiseHostName</td>
<td>Optional; specifies the local hostname address that will be advertised by the Prime Access Registrar server to other peers during CER/CEA exchange.</td>
</tr>
<tr>
<td>AdvertiseRealm</td>
<td>Advertising realm.</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Mandatory time interval, in milliseconds, to reactivate an inactive server.</td>
</tr>
<tr>
<td>Vendor</td>
<td>Select a valid vendor.</td>
</tr>
<tr>
<td>LimitOutstandingRequests</td>
<td>Check this box to limit the number of outstanding requests. If you enable this option, the number of outstanding requests for the Diameter remote server is limited to the value specified in the MaxOutstandingRequests field.</td>
</tr>
<tr>
<td>MaxOutstandingRequests</td>
<td>Maximum number of outstanding requests allowed for the Diameter remote server</td>
</tr>
<tr>
<td>MaxPendingPackets</td>
<td>Maximum number of packets that can be pending for the Diameter remote server.</td>
</tr>
<tr>
<td>DestinationRealm</td>
<td>Required. Destination realm to send Diameter packets to the remote server. The role of the remote server should be Relay.</td>
</tr>
<tr>
<td>TLS-Enabled</td>
<td>Check this box to enable TLS security mechanism for the Diameter remote server.</td>
</tr>
<tr>
<td>MaxTPSLimit</td>
<td>Maximum number of requests allowed per second for the Diameter remote server.</td>
</tr>
<tr>
<td>MaxSessionLimit</td>
<td>Maximum number of sessions allowed for the Diameter remote server.</td>
</tr>
</tbody>
</table>
### Table 3-65 Diameter Remote Server Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SCTPAdvHostName Section</strong></td>
<td>This section is available if the SCTP-Enabled option is checked.</td>
</tr>
<tr>
<td>Local</td>
<td>SCTP advertising host name of the local server.</td>
</tr>
<tr>
<td>Remote</td>
<td>SCTP advertising host name of the remote server.</td>
</tr>
<tr>
<td><strong>TLSEnabled Section</strong></td>
<td>This section is available if the TLS-Enabled option is checked.</td>
</tr>
<tr>
<td>PrivateKeyPassword</td>
<td>The password used to protect the server’s private key.</td>
</tr>
<tr>
<td>ServerKeyFile</td>
<td>The full pathname of the file containing the server’s RSA private key. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The valid encoding prefix is “PEM”. If an encoding prefix is not present, the file is assumed to be in PEM format. The following example assumes that the subdirectory pki under /cisco-ar contains the server’s certificate file. The file server-key.pem is assumed to be in PEM format. The file extension .pem is not significant.</td>
</tr>
<tr>
<td>set ServerKeyFile PEM:/cisco-ar/pki/server-key.pem</td>
<td></td>
</tr>
<tr>
<td>ServerCertificateFile</td>
<td>The full pathname of the file containing the server’s certificate or certificate chain used during the TLS exchange. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The valid encoding prefix is PEM. If an encoding prefix is not present, the file is assumed to be in PEM format.</td>
</tr>
<tr>
<td>CACertificateFile</td>
<td>The full pathname of the file containing trusted CA certificates used for client verification. The file can contain more than one certificate, but all certificates must be in PEM format. DER encoding is not allowed.</td>
</tr>
<tr>
<td>CACertificatePath</td>
<td>The name of a directory containing trusted CA certificates (in PEM format) used for client verification. This parameter is optional, and if it is used there are some special preparations required for the directory it references. Each certificate file in this directory must contain exactly one certificate in PEM format. The server looks up the certificate files using the MD5 hash value of the certificate’s subject name as a key. The directory must therefore also contain a set of symbolic links each of which points to an actual certificate file. The name of each symbolic link is the hash of the subject name of the certificate. For example, if a certificate file named ca-cert.pem is located in the CACertificatePath directory, and the MD5 hash of the subject name contained in ca-cert.path.pem is 1b96dd93, then a symbolic link named 1b96dd93 must point to ca-cert.pem. If there are subject name collisions such as multiple certificates with the same subject name, each link name must be indexed with a numeric extension as in 1b96dd93.0 and 1b96dd93.1.</td>
</tr>
</tbody>
</table>
You can use the Domain Authentication-RemoteServers page for the following:
- Filtering Records
- ODBC/OCI
- Editing Records
- Deleting Records

Adding Diameter Remote Server Details

To add new Diameter remote server details:

**Step 1** Choose **Network Resources > RemoteServers > Diameter**. The Diameter-Remote Servers page is displayed.

**Step 2** Click **Add** to add Diameter remote server details.

**Step 3** Enter the required details as described in Table 3-65.

**Step 4** Click **Add Diameter Server** to save the details. Click **Cancel** to return to the previous page without saving the details.

The Diameter-Remote Servers page is displayed with the newly added details or a respective error message is displayed.

Others

This feature of GUI allows you to set other specifications. The various types of protocols are:
- Radius
Table 3-66 lists and describes the fields in the Remote Server Details page. The fields listed below are the entire list of all the available protocols. The fields are displayed based on the type of protocol selected.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Remote Server Details</strong></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Required; name of the server.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the server.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Required; type of the remote server. Choose from one of the following options:</td>
</tr>
<tr>
<td></td>
<td>• Radius</td>
</tr>
<tr>
<td></td>
<td>• Dynamic DNS</td>
</tr>
<tr>
<td></td>
<td>• Map-Gateway</td>
</tr>
<tr>
<td></td>
<td>• Prepaid-CRB</td>
</tr>
<tr>
<td></td>
<td>• Prepa-IS835C</td>
</tr>
<tr>
<td></td>
<td>• Sigtran</td>
</tr>
<tr>
<td></td>
<td>• Sigtran-m3ua</td>
</tr>
<tr>
<td>IP Address</td>
<td>Required; this property specifies where to send the proxy request. The</td>
</tr>
<tr>
<td></td>
<td>IP is the address of the remote server. You must set it to a valid IP</td>
</tr>
<tr>
<td></td>
<td>address.</td>
</tr>
<tr>
<td>Port</td>
<td>By default, Prime Access Registrar listens on ports 1812 and 1813.</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Mandatory time interval (in milliseconds) to activate an inactive server;</td>
</tr>
<tr>
<td></td>
<td>defaults to 300000 ms.</td>
</tr>
<tr>
<td>MaxTries</td>
<td>Number of times the server tries to send dynamic updates to a server.</td>
</tr>
<tr>
<td>Initial Timeout</td>
<td>Time, in milliseconds, that the server waits for a response before</td>
</tr>
<tr>
<td></td>
<td>retrying a request.</td>
</tr>
<tr>
<td>SharedSecret</td>
<td>Required; the secret shared between the remote server and the RADIUS server.</td>
</tr>
<tr>
<td>Vendor</td>
<td>Optional; when set, must be the name of a known Vendor.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; when set, must be the name of a known incoming script. Prime</td>
</tr>
<tr>
<td></td>
<td>Access Registrar runs the IncomingScript after it receives the response.</td>
</tr>
</tbody>
</table>
### Table 3-66 Other Server Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OutGoingScript</td>
<td>Optional; when set, must be the name of a known outgoing script. Prime Access Registrar runs the OutgoingScript just before it sends the proxy request to the remote server.</td>
</tr>
<tr>
<td>AccountingPort</td>
<td>Port where the RADIUS server sends accounting packets.</td>
</tr>
</tbody>
</table>
| AcknowledgeAccounting| When ACKAccounting is TRUE, the Prime Access Registrar server waits for the Accounting-Response from the remote RADIUS server before sending the corresponding Accounting-Response to the client.  
When ACKAccounting is FALSE, the Prime Access Registrar server returns an Accounting-Response to the client without waiting for a response from the remote server. |
| SendandForget       | This field is available if the AcknowledgeAccounting option is disabled.  
After forwarding a proxy packet to the remote server and an initial response to the client, Prime Access Registrar maintains a buffer of the original request and a copy of the proxy request until it receives a response from the remote server or packet timeout is triggered.  
If SendandForget is enabled, Prime Access Registrar deletes the original and proxy requests from the buffer after sending the response to the client. This helps in reducing buffer pool exhaustion in case of a low-responding remote server. |
| Accept Dynamic Authorization Requests | The value is set to False, by default.                                                                                                           |
| MaxRename Retries   | Number of times that the resource managers can try to add a host even if it detects that the host's name is already present. This controls the number of times Prime Access Registrar tries to modify a host's name to resolve a conflict on each failed update. |
| MaxTPSLimit          | Maximum number of requests allowed per second for the remote server.  
This field is available only for RADIUS remote server.                                                                                           |
| MaxSessionLimit      | Maximum number of sessions allowed for the remote server.  
This field is available only for RADIUS and Sigtran-m3ua remote servers.                                                                           |
| Trim HostName        | Controls whether Prime Access Registrar trims the hostname string to the first period character. If this attribute is enabled, the hostname is truncated before the period. If disabled, the server retains the period characters in the hostname. |
| FwdZoneTSIG          | Server-wide security key to process all forward zone dynamic DNS updates. This is used if a ForwardZoneTSIGKey was not specified on the Resource Manager.                                                        |
| ReverseZoneTSIG      | Server-wide security key to process all reverse zone dynamic DNS updates. This is used if a ReverseZoneTSIGKey was not specified on the Resource Manager.                                                       |
### Table 3-66  Other Server Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File Name</td>
<td>Name of the shared library provided by the billing server vendor, such as <code>libprepaid.so</code></td>
</tr>
<tr>
<td>Connections</td>
<td>Number of threads the prepaid service and billing server can each use (default is 8).</td>
</tr>
<tr>
<td>HostName</td>
<td>Required; hostname of the remote server.</td>
</tr>
<tr>
<td>Local Sub System Number</td>
<td>Required; the default value for this property is 0. This represents the subsystem number used by SUA user.</td>
</tr>
<tr>
<td>CgPA Global Title Address</td>
<td>Required; represents the Global Title Address of CallingPartyAddress.</td>
</tr>
<tr>
<td>Set OPC In CgPA</td>
<td>Required; if it is set to TRUE, OPC will be used in CallingPartyAddress.</td>
</tr>
<tr>
<td>CdPANumberingPlan</td>
<td>Required; used to specify the numbering plan of the called party. The default value is 7.</td>
</tr>
<tr>
<td>CgPANumberingPlan</td>
<td>Required; used to specify the numbering plan of the calling party. The default value is 7.</td>
</tr>
<tr>
<td>Global Title Translation Script</td>
<td>This is used to specify the name of script which is responsible for translating IMSI to GTA.</td>
</tr>
<tr>
<td>SUA Configuration Filename</td>
<td>Required; used to specify the name of configuration file for SUA stack initialization.</td>
</tr>
<tr>
<td>Max Outstanding Requests</td>
<td>This represents the maximum outstanding request to HLR.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Required; represents the how long the remote server should wait before marking the request as timeout.</td>
</tr>
<tr>
<td>Limit Outstanding Requests</td>
<td>Limits the outstanding request to HLR when it is set to TRUE.</td>
</tr>
<tr>
<td>SourceIPAddress</td>
<td>Required; name of the local IP address.</td>
</tr>
<tr>
<td>SourcePort</td>
<td>Required; specify the port number in which Prime Access Registrar is installed for M3UA transactions.</td>
</tr>
<tr>
<td>LocalSubSystemNumber</td>
<td>Required; the local sub system number is set as 149 by default.</td>
</tr>
<tr>
<td>DestinationPort</td>
<td>Required; specify the destination port number to which Prime Access Registrar connects.</td>
</tr>
<tr>
<td>IMSITranslationScript</td>
<td>Specify the scripting point that is used to modify the IMSI based on the requirement before sending the request to STP/HLR.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Required; specify the time (in seconds) to wait before an authentication request times out; defaults to 120.</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Required; specify the time interval (in milliseconds) to activate an inactive server; defaults to 300000 ms (which is 5 minutes).</td>
</tr>
</tbody>
</table>
Network Resources

### Chapter 3 Using the Graphical User Interface

#### Network Resources

Prime Access Registrar uses this property in conjunction with the MaxOutstandingRequests property to tune the RADIUS server's use of the HLR. The default is FALSE.

When you set this property to TRUE, the number of outstanding requests for this RemoteServer is limited to the value you specified in MaxOutstandingRequests. When the number of requests exceeds this number, Prime Access Registrar queues the remaining requests, and sends them as soon as the number of outstanding requests drops to this number.

**MaxOutstandingRequests** Required; specify the maximum number of outstanding requests allowed for this remote server.

**MAP-Version** Required; specify the MAP version as 2 or 3 that HLR supports.

**NetworkVariant** Required for SIGTRAN-M3UA remote server; Choose **ITU** or **ANSI** to represent the standard that SIGTRAN-M3UA remote server supports.

**SubServiceField** Required; specify the type of network to which this SAP belongs. The possible options are INT and NAT which represents international network and national network respectively.

**TCAPVariant** Required; specify the name of the TCAP network variant switch. The possible options are ITU88, ITU92, or ITU96.

**NetworkAppearance** Required; specify the network appearance code which ranges from 0-2147483647.

This field is optional for SIGTRAN-M3UA remote servers as per the RFC 4666 ([http://tools.ietf.org/html/rfc4666](http://tools.ietf.org/html/rfc4666)). You can set the value to 0 to remove network appearance from the data packet.

**NetworkIndicator** Required; specify the network indicator used in SCCP address. The possible options are NAT and INT which represents international network and national network respectively.

**RoutingIndicator** Required; specify the routing indicator. The possible options are RTE_GT or RTE_SSN which is used to route the packets for HLR.

**MLCNumber** Required; specify the MLC number which is required for M3UA service for fetching the MSISDN from the HLR. This is the map layer network node number by which the HLR identifies the Prime Access Registrar in the network. The MLC number is configured in E.164 format.

**Note** MLC is a max-15 digit number.

**TrafficMode** Required; specify the traffic mode values for the HLR.

**LoadShareMode** Required; specify the load share mode for the HLR.

When there is more than one associations with HLR, then the load sharing is set as Signaling Link Selection (SLS). SLS is done based on a simple round-robin basis.

### Table 3-66 Other Server Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limit Outstanding Requests</td>
<td>Prime Access Registrar uses this property in conjunction with the MaxOutstandingRequests property to tune the RADIUS server's use of the HLR. The default is FALSE. When you set this property to TRUE, the number of outstanding requests for this RemoteServer is limited to the value you specified in MaxOutstandingRequests. When the number of requests exceeds this number, Prime Access Registrar queues the remaining requests, and sends them as soon as the number of outstanding requests drops to this number.</td>
</tr>
<tr>
<td>MaxOutstandingRequests</td>
<td>Required; specify the maximum number of outstanding requests allowed for this remote server.</td>
</tr>
<tr>
<td>MAP-Version</td>
<td>Required; specify the MAP version as 2 or 3 that HLR supports.</td>
</tr>
<tr>
<td>NetworkVariant</td>
<td>Required for SIGTRAN-M3UA remote server; Choose <strong>ITU</strong> or <strong>ANSI</strong> to represent the standard that SIGTRAN-M3UA remote server supports.</td>
</tr>
<tr>
<td>SubServiceField</td>
<td>Required; specify the type of network to which this SAP belongs. The possible options are INT and NAT which represents international network and national network respectively.</td>
</tr>
<tr>
<td>TCAPVariant</td>
<td>Required; specify the name of the TCAP network variant switch. The possible options are ITU88, ITU92, or ITU96.</td>
</tr>
<tr>
<td>NetworkAppearance</td>
<td>Required; specify the network appearance code which ranges from 0-2147483647. This field is optional for SIGTRAN-M3UA remote servers as per the RFC 4666 (<a href="http://tools.ietf.org/html/rfc4666">http://tools.ietf.org/html/rfc4666</a>). You can set the value to 0 to remove network appearance from the data packet.</td>
</tr>
<tr>
<td>NetworkIndicator</td>
<td>Required; specify the network indicator used in SCCP address. The possible options are NAT and INT which represents international network and national network respectively.</td>
</tr>
<tr>
<td>RoutingIndicator</td>
<td>Required; specify the routing indicator. The possible options are RTE_GT or RTE_SSN which is used to route the packets for HLR.</td>
</tr>
<tr>
<td>MLCNumber</td>
<td>Required; specify the MLC number which is required for M3UA service for fetching the MSISDN from the HLR. This is the map layer network node number by which the HLR identifies the Prime Access Registrar in the network. The MLC number is configured in E.164 format.</td>
</tr>
</tbody>
</table>

Note: MLC is a max-15 digit number.
### Table 3-66 Other Server Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCCPVariant</td>
<td>The Signaling Connection Control Part (SCCP) variant of the Global Title:</td>
</tr>
<tr>
<td></td>
<td>- Select <strong>ITU88</strong>, <strong>ITU92</strong>, or <strong>ITU96</strong>, if NetworkVariant is set to ITU.</td>
</tr>
<tr>
<td></td>
<td>- Select <strong>ANS88</strong>, <strong>ANS92</strong>, or <strong>ANS96</strong>, if NetworkVariant is set to ANS.</td>
</tr>
<tr>
<td>MaxTimeOuts</td>
<td>Maximum number of timeouts allowed for the remote server.</td>
</tr>
<tr>
<td><strong>Routing Parameters</strong></td>
<td></td>
</tr>
<tr>
<td>OriginPointCode</td>
<td>Required; specify the originating point of a message in a Signalling network. The value ranges from 0 - 16777215.</td>
</tr>
<tr>
<td>DestinationPointCode</td>
<td>Required; specify the destination address of a Signalling point in a SS7 network.</td>
</tr>
<tr>
<td>RemoteSubSystemNumber</td>
<td>Required; specify the sub system number of the remote server. The RemoteSubSystemNumber is set as 6 by default.</td>
</tr>
<tr>
<td>OPCMask</td>
<td>Required; specify the wild card mask for the origin point code. The value ranges from 0 - 16777215.</td>
</tr>
<tr>
<td>DPCMask</td>
<td>Specify the wild card mask for the destination point code. The value ranges from 0 - 16777215.</td>
</tr>
<tr>
<td>ServiceIndicatorOctet</td>
<td>Specify the service identifier octet. The value ranges from 0 - 255.</td>
</tr>
<tr>
<td>RoutingContext</td>
<td>Required; specify the routing context which ranges from 0 - 16777215.</td>
</tr>
<tr>
<td><strong>Source &amp; Destination IP Addresses</strong></td>
<td></td>
</tr>
<tr>
<td>SourceIPAddresses</td>
<td>Applicable only for Sigtran-m3ua protocol type. Enter the source IP address to be configured on the remote server and then click <strong>Add</strong>. The entered IP address is displayed in the SourceIPAddresses list box. Click <strong>Delete</strong> to remote the IP address from the list.</td>
</tr>
<tr>
<td>DestinationIPAddresses</td>
<td>Applicable only for Sigtran-m3ua protocol type. Enter the destination IP address to be configured on the remote server and then click <strong>Add</strong>. The entered IP address is displayed in the DestinationIPAddresses list box. Click <strong>Delete</strong> to remote the IP address from the list.</td>
</tr>
</tbody>
</table>

You can use the RemoteServers page allows for the following:

- **Filtering Records**
- **Setting Other Specifications**
- **Editing Records**
- **Deleting Records**

### Setting Other Specifications

To set up other specifications:

**Step 1** Select **Network Resources > RemoteServers > Others**. The RemoteServers page is displayed.
Click **Add** to add other specifications. The Remote Server Details page is displayed.

**Step 3**
Enter the required details.

**Step 4**
Click **Add Radius Server** to save the specified details in the Remote Server Details page. Otherwise click **Cancel** to return to the RemoteServers page without saving the details.

The RemoteServers page is displayed with the newly added details or a respective error message is displayed.

---

## Administration

Administration constitutes the maintenance and management of details specific administrator, various statistical data respective to the administrators, backing up and restoring server details, and license management of the server.

This section describes the following:

- Administrators
- Statistics
- DiameterStatistics
- TACACSStatistics
- Back Up and Restore
- License Upload

## Administrators

Prime Access Registrar provided *super-user* administrative access in which administrator can perform all tasks including starting and stopping the system and changing the configuration.

Prime Access Registrar also provides view-only administrative access. View-only access restricts an administrator to only being able to observe the system and prevents that user from making changes.

Table 3-67 lists and describes the fields in the Administrator Details page.

### Table 3-67 Administrator Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; administrator's user ID.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the administrator.</td>
</tr>
<tr>
<td>New Password</td>
<td>Required; encrypted password of the administrator.</td>
</tr>
<tr>
<td>Confirm New Password</td>
<td>Required; encrypted password of the administrator and must match Password.</td>
</tr>
<tr>
<td>View Only</td>
<td>Default value (FALSE) indicates that the administrator is able to modify the configuration. When set to TRUE, the administrator can only view the server configuration and set the change the server trace level.</td>
</tr>
</tbody>
</table>
You can use the Administrators page for the following:

- Filtering Records
- Adding Administrator Details
- Statistics
- Editing Records
- Deleting Records

Adding Administrator Details

To add new Administrator details:

**Step 1** Choose **Administration > Administrators**. The Administrators page is displayed.

**Step 2** Click **Add** to add administrator details. The Administrator Details page is displayed.

**Step 3** Specify the required details.

**Step 4** Click **Submit** to save the specified details in the Administrator Details page. Otherwise click **Cancel** to return to the Administrators page without saving the details.

The Administrators page is displayed with the newly added details or a respective error message is displayed.

Statistics

This feature provides statistical information on the specified RADIUS server. Table 3-68 lists the statistics information of the RADIUS server.

<table>
<thead>
<tr>
<th>Stats Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverStartTime</td>
<td>Indicates the start time of the server.</td>
</tr>
<tr>
<td>serverResetTime</td>
<td>Indicates the time when the server was reloaded.</td>
</tr>
<tr>
<td>serverStat</td>
<td>Indicates if the server is running or stopped.</td>
</tr>
<tr>
<td>totalPacketsInPool</td>
<td>Number of packets that can be accommodated in the pool.</td>
</tr>
<tr>
<td>totalPacketsReceived</td>
<td>Number of packets that are received by RADIUS server.</td>
</tr>
<tr>
<td>totalPacketsSent</td>
<td>Number of packets that are sent by RADIUS server.</td>
</tr>
<tr>
<td>totalRequests</td>
<td>Number of requests received by RADIUS server. This includes access requests and accounting requests.</td>
</tr>
</tbody>
</table>
Table 3-68  aregcmd stats Information for RADIUS server (continued)

<table>
<thead>
<tr>
<th>Stats Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>totalResponses</td>
<td>Number of responses sent by RADIUS server. This includes access accepts/rejects and accounting responses.</td>
</tr>
<tr>
<td>totalAccessRequests</td>
<td>Number of access requests received/processed by RADIUS server.</td>
</tr>
<tr>
<td>totalAccessAccepts</td>
<td>Number of access accepts sent by RADIUS server.</td>
</tr>
<tr>
<td>totalAccessChallenges</td>
<td>Number of access challenges sent by RADIUS server.</td>
</tr>
<tr>
<td>totalAccessRejects</td>
<td>Number of access rejects sent by RADIUS server.</td>
</tr>
<tr>
<td>totalAccessResponses</td>
<td>Number of access responses sent by RADIUS server.</td>
</tr>
<tr>
<td>totalAccountingRequests</td>
<td>Number of accounting requests received by RADIUS server.</td>
</tr>
<tr>
<td>totalAccountingResponses</td>
<td>Number of accounting responses sent by RADIUS server.</td>
</tr>
<tr>
<td>totalStatusServerRequests</td>
<td>Number of status server request received by RADIUS server.</td>
</tr>
<tr>
<td>totalAscendIPAAllocateRequests</td>
<td>Number of requests received related to Ascend IP address allocation.</td>
</tr>
<tr>
<td>totalAscendIPAAllocateResponses</td>
<td>Number of responses sent related to Ascend IP Address Allocation.</td>
</tr>
<tr>
<td>totalAscendIPAReleaseRequests</td>
<td>Number of requests received related to Ascend IP Address release.</td>
</tr>
<tr>
<td>totalAscendIPAReleaseResponses</td>
<td>Number of responses sent related to Ascend IP Address release.</td>
</tr>
<tr>
<td>totalUSRNASRebootRequests</td>
<td>Number of user NAS reboot request received by RADIUS server.</td>
</tr>
<tr>
<td>totalUSRNASRebootResponses</td>
<td>Number of user NAS reboot response sent by RADIUS server.</td>
</tr>
<tr>
<td>totalUSRResourceFreeRequests</td>
<td>Number of user resource free request received by RADIUS server.</td>
</tr>
<tr>
<td>totalUSRResourceFreeResponses</td>
<td>Number of user resource free response sent by RADIUS server.</td>
</tr>
<tr>
<td>totalUSRQueryResourceRequests</td>
<td>Number of user query resource request received by RADIUS server.</td>
</tr>
<tr>
<td>totalUSRQueryResourceResponses</td>
<td>Number of user query resource response sent by RADIUS server.</td>
</tr>
<tr>
<td>totalUSRQueryReclaimRequests</td>
<td>Number of user query reclaim request received by RADIUS server.</td>
</tr>
</tbody>
</table>
### Table 3-68  
aregcmd stats Information for RADIUS server (continued)

<table>
<thead>
<tr>
<th>Stats Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>totalUSRQueryReclaimResponses</td>
<td>Number of user query reclaim response sent by RADIUS server.</td>
</tr>
<tr>
<td>totalPacketsInUse</td>
<td>Number of packets that are being used.</td>
</tr>
<tr>
<td>totalPacketsDrained</td>
<td>Number of packets that are drained.</td>
</tr>
<tr>
<td>totalPacketsDropped</td>
<td>Number of packets that are dropped.</td>
</tr>
<tr>
<td>totalPayloadDecryptionFailures</td>
<td>Number of failures due to payloads decryption.</td>
</tr>
</tbody>
</table>

The following fields appear when you select a RADIUS client from the Clients drop-down list box at the bottom of the page.

- **RADIUS Client statistics for:** provides client’s IP address, name, and IP address type.
- **totalAuthAccessRequests** : Number of authentication access requests that are received by Prime Access Registrar from the client.
- **totalAuthDupAccessRequests** : Number of duplicate authentication access requests that are received by Prime Access Registrar from the client.
- **totalAuthAccessAccepts** : Number of authentication access requests from the client that are accepted by Prime Access Registrar.
- **totalAuthAccessRejects** : Number of authentication access requests from the client that are rejected by Prime Access Registrar.
- **totalAuthAccessChallenges** : Number of authentication challenges that are faced by Prime Access Registrar for the requests raised by the client.
- **totalAuthMalformedAccessRequests** : Number of malformed authentication access requests that are received by Prime Access Registrar from the client.
- **totalAuthBadAuthenticators** : Number of bad authentication access requests that are received by Prime Access Registrar from the client.
- **totalAuthPacketsDropped** : Number of authentication access requests received from the client that are dropped by Prime Access Registrar. The packets, which are invalid and do not fulfill the parsing conditions, are dropped.
- **totalAuthUnknownTypes** : Number of unknown authentication access requests that are received by Prime Access Registrar from the client.
Resetting Server Statistics

To reset server statistics:

**Step 1** Choose Administration > Statistics. The Radius Server Statistics page is displayed.

**Step 2** Click Reset to reset all the RADIUS server statistics.

### DiameterStatistics

Prime Access Registrar supports statistic of Diameter messages through the CLI/GUI and SNMP. The existing ‘stats’ module has been extended to include additional counters related to Diameter. The Diameter statistics includes peer statistics and global summary statistics details on the specified server.
Table 3-69 lists the statistics information of the Diameter server. The statistical information in Table 3-70 is displayed based on the Diameter peer selected. Table 3-71 is displayed based on the Diameter remote server selected.

**Table 3-69 Diameter Stats Information**

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Diameter Statistics</strong></td>
<td></td>
</tr>
<tr>
<td>serverStartTime</td>
<td>The start time of the server.</td>
</tr>
<tr>
<td>serverResetTime</td>
<td>The reset time of the server.</td>
</tr>
<tr>
<td>serverState</td>
<td>The state of the server.</td>
</tr>
<tr>
<td>cdbpLocalStatsTotalUpTime</td>
<td>The total time for which the Diameter server is up.</td>
</tr>
<tr>
<td>cdbpLocalResetTime</td>
<td>The time elapsed since a server was reset.</td>
</tr>
<tr>
<td>cdbpLocalStatsTotalPacketsIn</td>
<td>The total number of packets received by a Diameter Base protocol.</td>
</tr>
<tr>
<td>cdbpLocalStatsTotalPacketsOut</td>
<td>The total number of packets transmitted by a Diameter Base protocol.</td>
</tr>
<tr>
<td>cdbpLocalStatsTotalPacketsInUse</td>
<td>The total number of packets used.</td>
</tr>
<tr>
<td>Peer</td>
<td>The name of the peer. You can select a peer from the drop-down list.</td>
</tr>
</tbody>
</table>

**Table 3-70 Diameter Peer Stats Information**

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Diameter Peers:</strong> To view the following fields, select a Diameter peer from the Peer drop-down list box and then click <strong>Show Peer Stats</strong>. Click <strong>Reset</strong>, to reset all the Diameter statistics of the peer.</td>
<td></td>
</tr>
<tr>
<td>Stats for the Remote Server</td>
<td>The name of the selected peer.</td>
</tr>
<tr>
<td>ipaddress</td>
<td>The IP address of the peer.</td>
</tr>
<tr>
<td>port</td>
<td>The port of the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsState</td>
<td>Indicates the connection state in the Peer State Machine of the peer with which the Diameter server is communicating.</td>
</tr>
<tr>
<td>cdbpPeerStatsASAsOut</td>
<td>Number of Abort-Session-Answer messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsACRsIn</td>
<td>Number of Accounting-Request messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsACRsOut</td>
<td>Number of Accounting-Request messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsACAsIn</td>
<td>Number of Accounting-Answer messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsACAsOut</td>
<td>Number of Accounting-Answer messages that are sent to the peer.</td>
</tr>
</tbody>
</table>
### Table 3-70 Diameter Peer Stats Information (continued)

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdbpPeerStatsCERsIn</td>
<td>Number of Capabilities-Exchange-Request messages received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsCERsOut</td>
<td>Number of Capabilities-Exchange-Request messages sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsCEAsIn</td>
<td>Number of Capabilities-Exchange-Answer messages received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsCEAsOut</td>
<td>Number of Capabilities-Exchange-Answer messages sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsDWRsIn</td>
<td>Number of Device-Watchdog-Request messages received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsStateDuration</td>
<td>Represents the Peer state duration.</td>
</tr>
<tr>
<td>cdbpPeerStatsDWRsOut</td>
<td>Number of Device-Watchdog-Request messages sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsDWAsIn</td>
<td>Number of Device-Watchdog-Answer messages received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsDWAsOut</td>
<td>Number of Device-Watchdog-Answer messages sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsDPRsIn</td>
<td>Number of Disconnect-Peer-Request messages received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsDPRsOut</td>
<td>Number of Disconnect-Peer-Request messages sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsDPAsIn</td>
<td>Number of Disconnect-Peer-Answer messages received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsDPAsOut</td>
<td>Number of Disconnect-Peer-Answer messages sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsRARsIn</td>
<td>Number of Re-Auth-Request messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsRARsOut</td>
<td>Number of Re-Auth-Request messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsRAAsIn</td>
<td>Number of Re-Auth-Answer messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsRAAsOut</td>
<td>Number of Re-Auth-Answer messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsLastDiscCause</td>
<td>The last cause for a peer’s disconnection.</td>
</tr>
<tr>
<td>cdbpPeerStatsSTRsIn</td>
<td>Number of Session-Termination-Request messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsSTRsOut</td>
<td>Number of Session-Termination-Request messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsSTAsIn</td>
<td>Number of Session-Termination-Answer messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsSTAsOut</td>
<td>Number of Session-Termination-Answer messages that are sent to the peer.</td>
</tr>
</tbody>
</table>
### Table 3-70 Diameter Peer Stats Information (continued)

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdbpPeerStatsDWReqTimer</td>
<td>The interval between the packets that are sent to the peers.</td>
</tr>
<tr>
<td>cdbpPeerstatsRedirectEvents</td>
<td>Number of redirects that are sent from a peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsAccDupRequests</td>
<td>Number of duplicate Diameter Accounting-Request packets.</td>
</tr>
<tr>
<td>cdbpPeerStatsMalformedReqsts</td>
<td>Number of malformed Diameter packets that are received.</td>
</tr>
<tr>
<td>cdbpPeerStatsAccsNotRecorded</td>
<td>Number of Diameter Accounting-Request packets that are received and responded but not recorded.</td>
</tr>
<tr>
<td>cdbpPeerStatsWhoInitDisconnect</td>
<td>Indicates whether the host or peer initiated the disconnect.</td>
</tr>
<tr>
<td>cdbpPeerStatsAccRetrans</td>
<td>Number of Diameter Accounting-Request packets that are retransmitted to the Diameter server.</td>
</tr>
<tr>
<td>cdbpPeerStatsTotalRetrans</td>
<td>Number of Diameter packets that are retransmitted to the Diameter server. This does not include the Diameter Accounting-Request packets that are retransmitted.</td>
</tr>
<tr>
<td>cdbpPeerStatsAccPendReqstsOut</td>
<td>Number of Diameter Accounting-Request packets that are sent to the peer which have not yet timed out or received a response. This variable is incremented when an Accounting-Request is sent to the server and decremented due to receipt of an Accounting-Response, a timeout or a retransmission.</td>
</tr>
<tr>
<td>cdbpPeerStatsAccReqstsDropped</td>
<td>Number of Accounting-Requests to the server that are dropped.</td>
</tr>
<tr>
<td>cdbpPeerStatsHByHDropMessages</td>
<td>An answer message that is received with an unknown hop-by-hop identifier. This does not include the accounting requests that are dropped.</td>
</tr>
<tr>
<td>cdbpPeerStatsEToEDupMessages</td>
<td>The duplicate answer messages that are locally consumed. This does not include duplicate accounting requests that are received.</td>
</tr>
<tr>
<td>cdbpPeerStatsUnknownTypes</td>
<td>Number of Diameter packets of unknown type that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsProtocolErrors</td>
<td>Number of protocol errors that are returned to peer, but not including the redirects.</td>
</tr>
<tr>
<td>cdbpPeerStatsTransientFailures</td>
<td>Indicates the transient failure count.</td>
</tr>
<tr>
<td>cdbpPeerStatsPermanentFailures</td>
<td>Indicates the permanent failure count.</td>
</tr>
</tbody>
</table>
### Table 3-70 Diameter Peer Stats Information (continued)

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdbpPeerStatsDWCCurrentStatus</td>
<td>Indicates the connection status of the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsTransportDown</td>
<td>Number of unexpected transport failures.</td>
</tr>
<tr>
<td>cdbpPeerStatsTimeoutConnAtmpts</td>
<td>Number of times the server attempts to connect to a peer when there is no transport connection with the peer. This is reset on disconnection.</td>
</tr>
<tr>
<td>cdbpPeerStatsASRsIn</td>
<td>Number of Abort-Session-Request messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsASRsOut</td>
<td>Number of Abort-Session-Request messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsASAsIn</td>
<td>Number of Abort-Session-Answer messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsDERsIn</td>
<td>Number of Diameter-EAP-Request messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsDERsOut</td>
<td>Number of Diameter-EAP-Request messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsDEAsIn</td>
<td>Number of Diameter-EAP-Answer messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsDEAsOut</td>
<td>Number of Diameter-EAP-Answer messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsAARsIn</td>
<td>Number of AA-Request messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsAARsOut</td>
<td>Number of AA-Request messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsAAAsIn</td>
<td>Number of AA-Answer messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsAAAsOut</td>
<td>Number of AA-Answer messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsMARsIn</td>
<td>Number of Multimedia-Authentication-Request messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsMARsOut</td>
<td>Number of Multimedia-Authentication-Request messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsMAAsIn</td>
<td>Number of Multimedia-Authentication-Answer messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsMAAsOut</td>
<td>Number of Multimedia-Authentication-Answer messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsSARsIn</td>
<td>Number of Server-Assignment-Request messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsSARsOut</td>
<td>Number of Server-Assignment-Request messages that are sent to the peer.</td>
</tr>
</tbody>
</table>
### Table 3-70 Diameter Peer Stats Information (continued)

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdbpPeerStatsSAAsIn</td>
<td>Number of Server-Assignment-Answer messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsSAAsOut</td>
<td>Number of Server-Assignment-Answer messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsRTRsIn</td>
<td>Number of Registration-Termination-Request messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsRTRsOut</td>
<td>Number of Registration-Termination-Request messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsRTAsIn</td>
<td>Number of Registration-Termination-Answer messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsRTAsOut</td>
<td>Number of Registration-Termination-Answer messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsPPRsIn</td>
<td>Number of Push-Profile-Request messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsPPRsOut</td>
<td>Number of Push-Profile-Request messages that are sent to the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsPPAsIn</td>
<td>Number of Push-Profile-Answer messages that are received from the peer.</td>
</tr>
<tr>
<td>cdbpPeerStatsPPAsOut</td>
<td>Number of Push-Profile-Answer messages that are sent to the peer.</td>
</tr>
</tbody>
</table>

### Table 3-71 Diameter Remote Server Stats Information

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Diameter RemoteServers</strong>: To view the following fields, select a remote server from the Remote-Servers drop-down list box and then click Show RemoteServer Stats. Click Reset, to reset all the Diameter statistics of the remote server.</td>
<td></td>
</tr>
<tr>
<td>Stats for the Remote Server</td>
<td>The name of the selected remote server.</td>
</tr>
<tr>
<td>ipaddress</td>
<td>The IP address of the remote server.</td>
</tr>
<tr>
<td>port</td>
<td>The port of the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrActive</td>
<td>Indicates whether the server was active (not in a down state).</td>
</tr>
<tr>
<td>cDiaRemSvrRTTAverage</td>
<td>Average round trip time since the last server restart.</td>
</tr>
<tr>
<td>cDiaRemSvrRTTDeviation</td>
<td>Indicates a standard deviation of the RTTAverage.</td>
</tr>
<tr>
<td>cDiaRemSvrServerType</td>
<td>Indicates the remote server type.</td>
</tr>
<tr>
<td>cDiaRemSvrTotalRequestsPending</td>
<td>Number of requests currently queued.</td>
</tr>
<tr>
<td>cDiaRemSvrTotalRequestsOutstanding</td>
<td>Number of requests currently proxied that have not yet returned</td>
</tr>
<tr>
<td>Metric</td>
<td>Value</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>cDiaRemSvrTotalRequestsAcknowledged</td>
<td>Number of responses received since last server restart.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsState</td>
<td>Indicates the connection state of the Diameter remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsASRsIn</td>
<td>Number of Abort-Session-Request messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsASRsOut</td>
<td>Number of Abort-Session-Request messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsASAsIn</td>
<td>Number of Abort-Session-Answer messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsASAsOut</td>
<td>Number of Abort-Session-Answer messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsACRsIn</td>
<td>Number of Accounting-Request messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsACRsOut</td>
<td>Number of Accounting-Request messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsACAsIn</td>
<td>Number of Accounting-Answer messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsACAsOut</td>
<td>Number of Accounting-Answer messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsCERsIn</td>
<td>Number of Capabilities-Exchange-Request messages received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsCERsOut</td>
<td>Number of Capabilities-Exchange-Request messages sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsCEAsIn</td>
<td>Number of Capabilities-Exchange-Answer messages received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsCEAsOut</td>
<td>Number of Capabilities-Exchange-Answer messages sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDWRsIn</td>
<td>Number of Device-Watchdog-Request messages received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDWRsOut</td>
<td>Number of Device-Watchdog-Request messages sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDWAsIn</td>
<td>Number of Device-Watchdog-Answer messages received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDWAsOut</td>
<td>Number of Device-Watchdog-Answer messages sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDPRsIn</td>
<td>Number of Disconnect-Peer-Request messages received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDPRsOut</td>
<td>Number of Disconnect-Peer-Request messages sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDPAsIn</td>
<td>Number of Disconnect-Peer-Answer messages received by the remote server.</td>
</tr>
</tbody>
</table>
### Table 3-71 Diameter Remote Server Stats Information (continued)

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cDiaRemSvrStatsDPAsOut</td>
<td>Number of Disconnect-Peer-Answer messages sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsRARsIn</td>
<td>Number of Re-Auth-Request messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsRARsOut</td>
<td>Number of Re-Auth-Request messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsRAAsIn</td>
<td>Number of Re-Auth-Answer messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsRAAsOut</td>
<td>Number of Re-Auth-Answer messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsSTRsIn</td>
<td>Number of Session-Termination-Request messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsSTRsOut</td>
<td>Number of Session-Termination-Request messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsSTAsIn</td>
<td>Number of Session-Termination-Answer messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsSTAsOut</td>
<td>Number of Session-Termination-Answer messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsRedirectEvents</td>
<td>Number of redirects that are sent from the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsAccDupRequests</td>
<td>Number of duplicate Diameter Accounting-Request packets.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsMalformedRequests</td>
<td>Number of malformed Diameter packets that are received.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsAccsNotRecorded</td>
<td>Number of Diameter Accounting-Request packets that are received and responded but not recorded.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsWhoInitDisconnect</td>
<td>Indicates whether the host or remote server initiated the disconnect.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsAccRetrans</td>
<td>Number of Diameter Accounting-Request packets that are retransmitted by the Diameter remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsTotalRetrans</td>
<td>Number of Diameter packets that are retransmitted by the Diameter remote server. This does not include the Diameter Accounting-Request packets that are retransmitted.</td>
</tr>
</tbody>
</table>
### Table 3-71 Diameter Remote Server Stats Information (continued)

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cDiaRemSvrStatsAccPendRequestsOut</td>
<td>Number of Diameter Accounting-Request packets that are sent by the remote server which have not yet timed out or received a response. This variable is incremented when an Accounting-Request is sent by the server and decremented due to receipt of an Accounting-Response, a timeout or a retransmission.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsAccRequestsDropped</td>
<td>Number of Accounting-Requests that are dropped.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsHByHDropMessages</td>
<td>An answer message that is received with an unknown hop-by-hop identifier. This does not include the accounting requests that are dropped.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsEToEDupMessages</td>
<td>The duplicate answer messages that are locally consumed. This does not include duplicate accounting requests that are received.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsUnknownTypes</td>
<td>Number of Diameter packets of unknown type that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsProtocolErrors</td>
<td>Number of protocol errors that are returned by the remote server, but not including the redirects.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsTransientFailures</td>
<td>Indicates the transient failure count.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsPermanentFailures</td>
<td>Indicates the permanent failure count.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDWCURRENTStatus</td>
<td>Indicates the connection status of the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsTransportDown</td>
<td>Number of unexpected transport failures.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsTimeoutConnAtmpts</td>
<td>Number of times the remote server attempts to reconnect when there is no transport connection. This is reset on disconnection.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsMARsIn</td>
<td>Number of Multimedia-Authentication-Request messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsMARsOut</td>
<td>Number of Multimedia-Authentication-Request messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsMAAsIn</td>
<td>Number of Multimedia-Authentication-Answer messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsMAAsOut</td>
<td>Number of Multimedia-Authentication-Answer messages that are sent by the remote server.</td>
</tr>
</tbody>
</table>
### Table 3-71 Diameter Remote Server Stats Information (continued)

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cDiaRemSvrStatsSARsIn</td>
<td>Number of Server-Assignment-Request messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsSARsOut</td>
<td>Number of Server-Assignment-Request messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsSAAsIn</td>
<td>Number of Server-Assignment-Answer messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsSAAsOut</td>
<td>Number of Server-Assignment-Answer messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsRTRsIn</td>
<td>Number of Registration-Termination-Request messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsRTRsOut</td>
<td>Number of Registration-Termination-Request messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsRTAsIn</td>
<td>Number of Registration-Termination-Answer messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsRTAsOut</td>
<td>Number of Registration-Termination-Answer messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsPPRsIn</td>
<td>Number of Push-Profile-Request messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsPPRsOut</td>
<td>Number of Push-Profile-Request messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsPPAsIn</td>
<td>Number of Push-Profile-Answer messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsPPAsOut</td>
<td>Number of Push-Profile-Answer messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDERsIn</td>
<td>Number of Diameter-EAP-Request messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDERsOut</td>
<td>Number of Diameter-EAP-Request messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDEAsIn</td>
<td>Number of Diameter-EAP-Answer messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsDEAsOut</td>
<td>Number of Diameter-EAP-Answer messages that are sent by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsAARsIn</td>
<td>Number of AA-Request messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsAARsOut</td>
<td>Number of AA-Request messages that are sent by the remote server.</td>
</tr>
</tbody>
</table>

Cisco Prime Access Registrar 7.0.1 User Guide
Prime Access Registrar supports CISCO-AAA-SERVER-MIB to describe the statistics of TACACS+ protocol. This is supported through CLI/GUI and SNMP.

Table 3-71 lists the statistics information and the meaning of the values.

### Table 3-71 Diameter Remote Server Stats Information (continued)

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cDiaRemSvrStatsAAAsIn</td>
<td>Number of AA-Answer messages that are received by the remote server.</td>
</tr>
<tr>
<td>cDiaRemSvrStatsAAAsOut</td>
<td>Number of AA-Answer messages that are sent by the remote server.</td>
</tr>
</tbody>
</table>

### TACACS Statistics

Prime Access Registrar supports CISCO-AAA-SERVER-MIB to describe the statistics of TACACS+ protocol. This is supported through CLI/GUI and SNMP.

Table 3-72 lists the statistics information and the meaning of the values.

### Table 3-72 TACACS Stats Information

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TACACS Statistics</td>
<td></td>
</tr>
<tr>
<td>serverStartTime</td>
<td>The start time of the server.</td>
</tr>
<tr>
<td>serverResetTime</td>
<td>The reset time of the server.</td>
</tr>
<tr>
<td>serverState</td>
<td>The state of the server.</td>
</tr>
<tr>
<td>totalPacketsReceived</td>
<td>Number of packets that are received by a TACACS+ protocol irrespective of the type of Authentication and Accounting.</td>
</tr>
<tr>
<td>totalPacketsSent</td>
<td>Number of packets that are sent by a TACACS+ protocol irrespective of the type of Authentication and Accounting.</td>
</tr>
<tr>
<td>totalRequests</td>
<td>Number of packet requests that are received by a TACACS+ protocol irrespective of the type of Authentication and Accounting.</td>
</tr>
<tr>
<td>totalResponses</td>
<td>Number of packet responses that are sent by a TACACS+ protocol irrespective of the type of Authentication and Accounting.</td>
</tr>
<tr>
<td>totalAuthenticationRequests</td>
<td>Number of authentication requests that are received by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAuthenticationAccepts</td>
<td>Number of authentication requests that are accepted by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAuthenticationRejects</td>
<td>Number of authentication requests that are rejected by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAuthenticationChallenges</td>
<td>Number of authentication challenges that are faced by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAuthenticationResponses</td>
<td>Number of authentication responses that are sent by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAuthorizationRequests</td>
<td>Number of authorization requests that are received by Prime Access Registrar.</td>
</tr>
</tbody>
</table>
Chapter 3  Using the Graphical User Interface

To back up and restore the server details, Choose Administration > Backup & Restore. The Backup page is displayed with the list of recently backed up details of the server with the date and time. This option allows you to take a backup of the database, sessions, and scripts, and stores it in /cisco-ar/backup directory.

**Backup Server Details**
To back up the server details:

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Choose Administration &gt; Backup &amp; Restore. The Backup page is displayed.</td>
</tr>
<tr>
<td>Step 2</td>
<td>Click Backup to take a backup of the database, sessions, and scripts, and stores it in /cisco-ar/backup directory. The details will be backed up and appended to the backup list and displayed in the Backup page.</td>
</tr>
</tbody>
</table>

**Restoring Server Details**
To restore the backed-up server details:

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1</td>
<td>Choose Administration &gt; Backup &amp; Restore. The Backup page is displayed.</td>
</tr>
<tr>
<td>Step 2</td>
<td>Choose the record from the backup list.</td>
</tr>
</tbody>
</table>

**Table 3-72  TACACS Stats Information (continued)**

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>totalAuthorizationAccepts</td>
<td>Number of authorization requests that are accepted by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAuthorizationRejects</td>
<td>Number of authorization requests that are rejected by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAuthorizationResponses</td>
<td>Number of authorization responses that are sent by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAccountingRequests</td>
<td>Number of accounting requests that are received by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAccountingAccepts</td>
<td>Number of accounting requests that are accepted by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAccountingRejects</td>
<td>Number of accounting requests that are rejected by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalAccountingResponses</td>
<td>Number of accounting requests that are sent by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalPayloadDecryptionFailures</td>
<td>Number of packets that are not decrypted by Prime Access Registrar.</td>
</tr>
<tr>
<td>totalPacketsDropped</td>
<td>Number of packets that are dropped by Prime Access Registrar. The packets are dropped, which are invalid and do not fulfill the parsing conditions.</td>
</tr>
</tbody>
</table>
Chapter 3      Using the Graphical User Interface

Read-Only GUI

Step 3  Click Restore. The details of the selected back up file will be restored successfully.

LicenseUpload

Prime Access Registrar license information are uploaded using the Upload feature. To upload the license file:

**Uploading License File**

To upload the Prime Access Registrar license file:

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Choose Administration &gt; LicenseUpload. The Prime Access Registrar License-Upload page is displayed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 2</td>
<td>Click Browse to locate the license file. The File Upload dialog box is displayed.</td>
</tr>
<tr>
<td>Step 3</td>
<td>Choose the required file.</td>
</tr>
<tr>
<td>Step 4</td>
<td>Click Upload. The selected file will be uploaded in /cisco-ar/license directory.</td>
</tr>
</tbody>
</table>

**Note** You need to ensure that the license file that you want to upload should be in .lic format.

| Step 5 | Click Reset to clear the text in the Select the File field, if you want to clear the selected path. |

Read-Only GUI

Prime Access Registrar provides a read-only GUI that enables an administrator to observe the system but prevents that administrator from making changes.

When you configure a user to be an administrator, check the View-Only check box to limit the administrator to view-only operation. You can also use the CLI by setting the View-Only property to TRUE under /Administrator/admin_name.

When using the Read-Only GUI, the Configuration, Network Resources and Administration sections are displayed as same as a fully-enabled administrator. The details of these sections are displayed in text format and cannot be edited.
Cisco Prime Access Registrar Server Objects

This chapter describes the objects you use to configure and operate your Cisco Prime Access Registrar (Prime Access Registrar) RADIUS server.

Prime Access Registrar is configured and operated through a set of objects. These objects are arranged in a hierarchy, with some of the objects containing subobjects; just as in a UNIX file system, in which directories can contain subdirectories. All of the objects, except those that are merely lists, contain properties that define the attributes or behavior of the object.

This chapter describes the following Prime Access Registrar objects:

- **Radius** — root of the configuration hierarchy
- **UserLists** — contains individual UserLists, which in turn contain users
- **UserGroups** — contains individual UserGroups
- **Policies** — contains individual Policies
- **Clients** — contains individual Clients
- **Vendors** — contains individual Vendors
- **Scripts** — contains individual Scripts
- **Services** — contains individual Services
- **Session Managers** — contains individual Session Managers
- **Resource Managers** — contains individual Resource Managers
- **Profiles** — contains individual Profiles
- **Rules** — contains individual Rules
- **FastRules** — contains attributes to add, modify, and delete in the request, response, and environment dictionaries.
- **Translations** — contains individual Translations
- **TranslationGroups** — contains individual Translation Groups
- **Remote Servers** — contains individual RemoteServers
- **Advanced** — contains advanced properties, Ports, Interfaces, Reply Messages, and the Attribute dictionary
Radius

The **Radius** object is the root of the hierarchy. For each installation of the Cisco Prime Access Registrar server, there is one instance of the **Radius** object. You reach all other objects in the hierarchy from the **Radius**.

The following is a listing of the RADIUS server object:

```plaintext
[ //localhost/Radius ]
  Name = Radius
  Description =
  Version = 7.0.0
  IncomingScript = query
  OutgoingScript =
  DefaultAuthenticationService = local-users
  DefaultAuthorizationService = local-users
  DefaultAccountingService = local-file
  DefaultSessionService =
  DefaultSessionManager = mgr-rad
  UserLists/
  UserGroups/
  Policies/
  Clients/
  Vendors/
  Scripts/
  Services/
  SessionManagers/
  ResourceManagers/
  Profiles/
  Rules/
  Translations/
  TranslationGroups/
  RemoteServers/
  CommandSets/
  DeviceAccessRules/
  FastRules/
  Advanced/
  Replication/
```

Table 4-1 lists the **Radius** properties. You can set or change Radius properties using the Cisco Prime Access Registrar `aregcmd` commands.

---

**Note**

When a field is listed as required, it means a value must be supplied; that is, the value must be set. You can use the default (if it is supplied) or you can change it to something else, but you cannot unset it. You **must** supply values for the required fields and for which no defaults exist.

---

**Table 4-1**  
*Radius Properties*

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the list of servers in the cluster</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the server</td>
</tr>
<tr>
<td>Version</td>
<td>Optional; if there is a script, it is the first script</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Cisco Prime Access Registrar runs when it receives a request from any client and/or for any service</td>
</tr>
</tbody>
</table>
Table 4-1  Radius Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OutgoingScript</td>
<td>Optional; if there is a script, it is the last script Cisco Prime Access Registrar runs before it sends a response to any client</td>
</tr>
<tr>
<td>DefaultAuthenticationService</td>
<td>Optional; Cisco Prime Access Registrar uses this property when none of the incoming scripts sets the environment dictionary variable Authentication-Service</td>
</tr>
<tr>
<td>DefaultAuthorizationService</td>
<td>Optional; Cisco Prime Access Registrar uses this property when none of the incoming scripts sets the environment dictionary variable Authorization-Service</td>
</tr>
<tr>
<td>DefaultAccountingService</td>
<td>Optional; Cisco Prime Access Registrar uses this property when none of the incoming scripts sets the environment dictionary variable Accounting-Service</td>
</tr>
<tr>
<td>DefaultSessionService</td>
<td>Cisco Prime Access Registrar uses this property when none of the incoming scripts sets the environment dictionary variable Session-Service. This field is mandatory if you are upgrading to a later version of Prime Access Registrar.</td>
</tr>
<tr>
<td>DefaultSessionManager</td>
<td>Cisco Prime Access Registrar uses this property if none of the incoming scripts sets the environment dictionary variable Session-Manager. This field is mandatory if you are upgrading to a later version of Prime Access Registrar.</td>
</tr>
</tbody>
</table>

The remaining Cisco Prime Access Registrar objects are sub-objects of the **Radius** object.

---

UserLists

The **UserLists** object contains all of the individual UserLists, which in turn, contain the specific users stored within Cisco Prime Access Registrar. Cisco Prime Access Registrar references each specific UserList by **name** from a Service whose type is set to **local**. When Cisco Prime Access Registrar receives a request, it directs it to a Service. When the Service has its type property set to **local**, the Service looks up the user’s entry in the specific UserList and authenticates and/or authorizes the user against that entry.

**Note**

Usernames might not include the forward slash (/) character. If the Cisco Prime Access Registrar server receives an access request packet with a User-Name attribute containing a forward slash character and the Prime Access Registrar server uses an internal UserList to look up users, the server produces an error (AX_EINVAL) and might fail. If usernames require a forward slash, use a script to translate the slash to an acceptable, unused character.

You can have more than one UserList in the **UserLists** object. Therefore, use the **UserLists** object to divide your user community by organization. For example, you might have separate **UserLists** objects for Company A and B, or you might have separate **UserLists** objects for different departments within a company.
Using separate UserLists objects allows you to have the same name in different lists. For example, if your company has three people named Bob and they work in different departments, you could create a UserList for each department, and each Bob could use his own name. Using UserLists lets you avoid the problem of Bob1, Bob2, and so on.

If you have more than one UserList, you can have a script Cisco Prime Access Registrar can run in response to requests. The script chooses the Service, and the Service specifies the actual UserList which contains the user. The alternative is dynamic properties.

The subobjects are the Users listed by name. Table 4-2 lists the UserLists object properties.

### Table 4-2 UserLists Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in UserLists.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the UserList.</td>
</tr>
</tbody>
</table>

### Users

The Users object contains all of the information necessary to authenticate a user or authorize a user. Users in local UserLists can have multiple profiles. Table 4-3 lists the Users object properties.

### Table 4-3 Users Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the specific UserList.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the user.</td>
</tr>
<tr>
<td>Password</td>
<td>Required; length must be between 0-253 characters.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Required; default is TRUE, which means the user is allowed access. Set to FALSE to cause Cisco Prime Access Registrar to deny the user access.</td>
</tr>
<tr>
<td>Group (Overridden by User-Group)</td>
<td>Optional; when you set this to the name of a UserGroup, Cisco Prime Access Registrar uses the properties specified in that UserGroup to authenticate and/or authorize the user.</td>
</tr>
<tr>
<td>BaseProfile (Overridden by User-Profile)</td>
<td>Optional; when you set this to the name of a Profile and the service-Type is not equal to Authenticate Only, Cisco Prime Access Registrar adds the properties in the Profile to the Response dictionary as part of the authorization.</td>
</tr>
<tr>
<td>AuthenticationScript</td>
<td>Optional; when you set this property to the name of a script, you can use the script to perform additional authentication checks to determine whether to accept or reject the user.</td>
</tr>
<tr>
<td>AuthorizationScript</td>
<td>Optional; when you set this property to the name of a script, you can use the script to add, delete, or modify the attributes of the Response dictionary.</td>
</tr>
<tr>
<td>UserDefined1</td>
<td>Optional; you can use this property to store notational information which you can then use to filter the UserList. This property also sets the environment variable for UserDefined1.</td>
</tr>
</tbody>
</table>
HiddenAttributes Property

The HiddenAttributes property in the user object provides a concatenation of all user-level reply attributes. The Prime Access Registrar server uses the HiddenAttributes property to construct and populate a virtual attributes directory.

The HiddenAttributes property is, in fact, hidden. It is not displayed and cannot be set or modified using aregcmd, but when an administrator issues a save, all values from the user’s Attributes directory go into the HiddenAttributes property and the persistent storage.

The attributes are added in a replace-if-present-add-if-not manner as used in the UserGroup-Base-Profile and User-Base-Profile.

The order of application of the attributes is as follows:

- UserGroup Base Profile
- UserGroup Attributes
- User Base Profile
- User Attributes

UserGroups

The UserGroups objects allow you to maintain common authentication and authorization attributes in one location, and then have many users reference them. By having a central location for attributes, you can make modifications in one place instead of having to make individual changes throughout your user community.

For example, you can use several UserGroups to separate users by the services they use, such as a group specifying PPP and another for Telnet.

Table 4-4 lists the UserGroups properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the UserGroup list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the group.</td>
</tr>
<tr>
<td>BaseProfile</td>
<td>Optional; when you set this to the name of a Profile, Cisco Prime Access Registrar adds the properties in the Profile to the response dictionary as part of the authorization.</td>
</tr>
<tr>
<td>AuthenticationScript</td>
<td>Optional; when you set this property to the name of a Script, you can use the Script to perform additional authentication checks to determine whether to accept or reject the user.</td>
</tr>
<tr>
<td>AuthorizationScript</td>
<td>Optional; when you set this property to the name of a Script, you can use the Script to add, delete, or modify the attributes of the Response dictionary.</td>
</tr>
</tbody>
</table>
Policies

A Policy is a set of rules applied to an Access-Request. If you are using Policies, the first one that must be created is SelectPolicy.

Table 4-5 lists the properties required for a given Policy.

Table 4-5  Policies Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Policies list</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the Policy</td>
</tr>
<tr>
<td>Grouping</td>
<td>Optional grouping of rules</td>
</tr>
</tbody>
</table>

Clients

All NASs and proxy clients that communicate directly with Cisco Prime Access Registrar must have an entry in the Clients list. This is required because NAS and proxy clients share a secret with the RADIUS server which is used to encrypt passwords and to sign responses. Table 4-6 lists the Client object properties.

Table 4-6  RADIUS Client Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required and should match the Client identifier specified in the standard RADIUS attribute, NAS-Identifier. The name must be unique within the Clients list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the client.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Required; specifies the client protocol which can be Radius, Diameter, or Tacaas-and-Radius.</td>
</tr>
</tbody>
</table>
Chapter 4  Cisco Prime Access Registrar Server Objects

Table 4-6  RADIUS Client Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPAddress</td>
<td>Required; must be a valid IP address and unique in the Clients list. Cisco Prime Access Registrar uses this property to identify the Client that sent the request, either using the source IP address to identify the immediate sender or using the NAS-IP-Address or NAS-IPv6-Address attribute in the Request dictionary to identify the NAS sending the request through a proxy. When a range is configured for a Client’s IPAddress property, any incoming requests whose source address belongs to the range specified, will be allowed for further processing by the server. Similarly when a wildcard (an asterisk ‘<em>’ in this case) is specified, any incoming requests whose source address matches the wildcard specification will be allowed. In both the cases, the configured client properties like SharedSecret, and Vendor are used to process the requests. You can specify a range of IP addresses using a hyphen as in: 100.1.2.11-20 You can use an asterisk wildcard to match all numbers in an IP address octet as in: 100.1.2.</em> You can specify an IPAddress and a subnet mask together using Classless Inter-Domain Routing (CIDR) notation as in: 100.1.2.0/24 You can use the IPAddress property to set a base address and use the NetMask property to specify the number of clients in the subnet range. The IP address format is enhanced to support IPv6 apart from IPv4. You can use an asterisk wildcard to match all numbers in an IP address octet as in: 1124:1124:1124:1124:<em>:</em>:<em>:</em> Note  The IPv6 address must be in standard notation.</td>
</tr>
<tr>
<td>SharedSecret</td>
<td>Required; must match the secret configured in the Client.</td>
</tr>
<tr>
<td>Type</td>
<td>Required; accept the default (NAS), or set it to ATM, Proxy, or NAS+Proxy.</td>
</tr>
<tr>
<td>Vendor</td>
<td>Optional; you can use this property when you need special processing for a specific vendor’s NAS. To use this property, you must configure a Vendor object and include a Script. Cisco Prime Access Registrar provides five Scripts you can use: one for Ascend, Cisco, Cabletron, Altiga, and one for USR. You can also provide your own Script.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; you can use this property to specify a Script you can use to determine the services to use for authentication, authorization, and/or accounting.</td>
</tr>
</tbody>
</table>
Table 4-6  RADIUS Client Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OutgoingScript</td>
<td>Optional; you can use this property to specify a Script you can use to make any Client-specific modifications when responding to a particular Client.</td>
</tr>
<tr>
<td>EnableDynamicAuthorization</td>
<td>Optional; when set to TRUE, this property enables Change of Authorization and Packet of Disconnect features.</td>
</tr>
<tr>
<td>DynamicAuthorizationServer</td>
<td>This subdirectory is only present in a client with EnableDynamicAuthorization set to TRUE and contains properties required for CoA and PoD requests.</td>
</tr>
<tr>
<td>Port</td>
<td>Located under the DynamicAuthorizationServer subdirectory, the default port is 3799.</td>
</tr>
<tr>
<td>InitialTimeout</td>
<td>Located under the DynamicAuthorizationServer subdirectory, the default is 5000.</td>
</tr>
<tr>
<td>MaxTries</td>
<td>Located under the DynamicAuthorizationServer subdirectory, the default is 3.</td>
</tr>
<tr>
<td>DynamicAuthSharedSecret</td>
<td>Located under the DynamicAuthorizationServer subdirectory, this is the shared secret used for communicating CoA and PoD packets with the client.</td>
</tr>
<tr>
<td>PODAttributeGroup</td>
<td>This property is found under the DynamicAuthorizationServer subdirectory and points to a group of attributes to be included in a POD request sent to this client. These attribute groups are created and configured under the AttributeGroups subdirectory in /Radius/Advanced.</td>
</tr>
<tr>
<td>COAAttributeGroup</td>
<td>This property is found under the DynamicAuthorizationServer subdirectory and points to a group of attributes to be included in a CoA request sent to this client. These attribute groups are created and configured under the AttributeGroups subdirectory in /Radius/Advanced.</td>
</tr>
<tr>
<td>NetMask</td>
<td>Specifies the subnet mask used with the network address setting configured for the IPAddress property when configuring a range of IP addresses.</td>
</tr>
<tr>
<td></td>
<td>This property is not used for a single client with an IP address only. The NetMask property is used to configure multiple clients when you configure a base IP address in the IPAddress property. You can set the NetMask property for a range of 256 clients using the following example: set NetMask 255.255.255.0</td>
</tr>
<tr>
<td></td>
<td>When the NetMask property indicates a pool of 256 address (255.255.255.0), the range of addresses reserved for clients is 0-255, as in 100.1.1.0-100.1.1.255.</td>
</tr>
<tr>
<td></td>
<td>Note  If you set the NetMask property, validation will fail if you attempt to specify a subnet mask using CIDR notation with the IPAddress property (described above).</td>
</tr>
</tbody>
</table>
**Table 4-6**  \textit{RADIUS Client Properties (continued)}

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| EnableNotifications       | Required; the default value is \text{FALSE} and indicates the client is not capable of receiving Accounting-Stop notifications from the Prime Access Registrar server.  
When set to \text{TRUE}, the client can receive Accounting-Stop notifications from the Prime Access Registrar server and additional properties must be configured under a new sub-directory named NotificationProperties. |
| NotificationProperties    | When the EnableNotifications property is set to \text{TRUE}, this subdirectory contains additional properties required to support the Query-Notify feature. |
| Port                      | Located under the NotificationProperties subdirectory, specifies the port used by the Prime Access Registrar server to receive Accounting-Stop packets. Required when EnableNotifications is set to \text{TRUE}; the default value is 1813. |
| InitialTimeout            | Located under the NotificationProperties subdirectory, specifies the timeout value in milliseconds the Prime Access Registrar server waits for an Accounting-Response packet before attempting a retry (sending another Accounting-Stop packet to the client). Required when EnableNotifications is set to \text{TRUE}; the default value is 5000. |
| MaxTries                  | Located under the NotificationProperties subdirectory, specifies the number of times the Prime Access Registrar server sends an Accounting-Stop packet to a client. Required when EnableNotifications is set to \text{TRUE}; the default value is 3. |
| NotificationAttributeGroup| Located under the NotificationProperties subdirectory, specifies the name of an attribute group under \text{/Radius/Advanced/AttributeGroups} that contains the attributes to be included when sending an the Accounting-Stop packet to this client. Required when EnableNotifications is set to \text{TRUE}; there is no default value. You must provide the name of a valid AttributeGroup and the named AttributeGroup must contain at least one valid attribute, or validation will fail. |
| EnforceTrafficThrottling  | Required; the default value is \text{TRUE} and indicates enforce traffic throttling for this client. This property is under \text{/Radius/Advanced/MaximumOutstanding/IncomingRequests}.  
When set to \text{FALSE}, the traffic throttling for the packet coming from this client is bypassed. |

Table 4-7 describes the Diameter client properties.
Vendors

The **Vendor** object provides a central location for specifying all of the request and response processing a particular NAS or Proxy vendor requires. Depending on the vendor, it might be necessary to map attributes in the request from one set to another, or to filter out certain attributes before sending the response to the client. For more information about standard RADIUS attributes, see Appendix C, “RADIUS Attributes.”

**Table 4-7 Diameter Client Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the client list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the client.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Required; specifies the client protocol which can be Radius or Diameter.</td>
</tr>
<tr>
<td>HostName</td>
<td>Required; hostname or IP address of the Diameter client.</td>
</tr>
<tr>
<td>Vendor</td>
<td>Optional; you can use this property when you need special processing for a specific vendor’s peer.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; specifies a script that you can use to make client-specific modifications when a request is received from a client.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; specifies a script that you can use to make any client-specific modifications when responding to a particular client.</td>
</tr>
<tr>
<td>Port</td>
<td>Required; port on which the client connects with Prime Access Registrar server.</td>
</tr>
<tr>
<td>SCTP-Enabled</td>
<td>Required, default value is False. If set to TRUE, SCTP will be used to establish the connection with the peer else TCP will be used.</td>
</tr>
</tbody>
</table>

When you have also set `/Radius/IncomingScript`, Cisco Prime Access Registrar runs that script before the vendor’s script. Conversely, when you have set a `/Radius/Outgoing` script, Cisco Prime Access Registrar runs the vendor’s script before that script.

**Table 4-8 lists the Vendor object properties.**

**Table 4-8 Vendor Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Vendors list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the vendor.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; when you specify an IncomingScript, Cisco Prime Access Registrar runs the script on all requests from clients that specify that vendor.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; when you specify an OutgoingScript, Cisco Prime Access Registrar runs the script on all responses to the Client.</td>
</tr>
</tbody>
</table>
Scripts

The **Script** objects define the function Cisco Prime Access Registrar invokes whenever the **Script** is referenced by name from other objects in the configuration.

You can write three types of scripts:

- **REX (RADIUS EXtension) scripts** are written in C or C++, and thus are compiled functions that reside in shared libraries.
- **Tcl scripts** are written in Tcl, and are interpreted functions defined in source files.
- **Java scripts**

**Note**

For more information about how to write scripts and how to incorporate them into Cisco Prime Access Registrar, see Chapter 11, “Using Extension Points.”

**Note**

Cisco is not liable for scripts developed by clients. See **Client Scripting** in Chapter 1, “Overview.”

Table 4-9 lists the **Script** object properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Scripts list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the script.</td>
</tr>
<tr>
<td>Language</td>
<td>Required; specify either REX, Tcl, or Java.</td>
</tr>
<tr>
<td>Filename</td>
<td>Required; specifies either a relative or absolute path. When you specify a relative path, the path must be relative to the $INSTALL/scripts/radius/$Language directory. When you specify an absolute path, the server must be able to reach it.</td>
</tr>
<tr>
<td>EntryPoint</td>
<td>Optional; when not set, Prime Access Registrar uses the value specified in the Name property.</td>
</tr>
<tr>
<td>InitEntryPoint</td>
<td>Optional; if set, it must be the name of the global symbol Prime Access Registrar should call when it initializes the shared library at system start up, and just before it unloads the shared library.</td>
</tr>
<tr>
<td>InitEntryPointArg</td>
<td>Optional; when set, it provides the arguments to be passed to the InitEntryPoint in the environmental variable Arguments.</td>
</tr>
<tr>
<td>ClassName</td>
<td>For Java language scripts, the name of the class that implements the extension interface; the .class file should be placed in /cisco-ar/scripts/radius/java</td>
</tr>
<tr>
<td>InitializeArg</td>
<td>Optional for Java language scripts; set to a string to be passed to the Initialize method if the class implements the optional ExtensionWithInitialization interface.</td>
</tr>
</tbody>
</table>

The **InitEntryPoint** properties allow you to perform initialization before processing and then cleanup before stopping the server. For example, when Prime Access Registrar unloads the script (when it stops the RADIUS server) it calls the **InitEntryPoint** again to allow it to perform any clean-up operations as a result of its initialization. One use of the function might be to allow the script to close an open Accounting log file before stopping the RADIUS server.
**Note**  
When you use a Prime Access Registrar file service, Prime Access Registrar automatically closes any opened files. However, if you write scripts that manipulate files, you are responsible for closing them.

**Note**  
If you have more than one extension point script (defined under `/Radius/Scripts`) using the same Java class, only one instance of the class is created and used for all the extension point scripts.

## Services

Cisco Prime Access Registrar supports authentication, authorization, and accounting (AAA) services. In addition to the variety of built-in AAA services (specified in the **Type** property), Cisco Prime Access Registrar also enables you to add new AAA services through custom shared libraries.

Table 4-10 lists the common **Services** properties. There are additional properties depending on the type of service.

### Table 4-10 Common Service Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Services list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the service.</td>
</tr>
<tr>
<td>Type</td>
<td>Required, must set it to a valid Prime Access Registrar service.</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; the default is <strong>DropPacket</strong>. This property defines how Cisco Prime Access Registrar handles requests if all servers listed in the <strong>RemoteServers</strong> properties are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: <strong>AcceptAll</strong>, <strong>DropPacket</strong>, or <strong>RejectAll</strong>.</td>
</tr>
<tr>
<td>OutageScript</td>
<td>Optional; if you set this property to the name of a script, Cisco Prime Access Registrar runs it when an outage occurs. This property allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
</tbody>
</table>

**Note**  
**OutagePolicy** also applies to Accounting-Requests. If an Accounting-Request is directed to an unavailable Service, then the values in Table 4-11 apply.
Table 4-11  OutagePolicy Request Packets

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Accounting-Request Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcceptAll</td>
<td>Continues processing the packet as if the Service was successful.</td>
<td>The Accounting-Request will continue through the server and a response will be sent.</td>
</tr>
<tr>
<td>DropPacket</td>
<td>Immediately drops the packet, no further processing, and does not send any response to the client for this packet.</td>
<td>The packet will be discarded and it will not be processed any further.</td>
</tr>
<tr>
<td>RejectAll</td>
<td>Rejects the packet, but continues processing it and sends the client a reject response.</td>
<td>The request will be dropped and no more processing will be done.</td>
</tr>
</tbody>
</table>

Types of Services

This section lists the types of services available in Prime Access Registrar with their required and optional properties. The service you specify determines what additional information you must provide.

This section contains the following topics:

- EAP Services
- File
- Group
- Java
- LDAP
- Local
- ODBC
- ODBC-Accounting
- Prepaid Services
- RADIUS
- Radius Query
- RADIUS-Session
- Rex
- WiMAX
- Diameter
- M3UA

EAP Services

Prime Access Registrar supports Extensible Authentication Protocol (EAP) and Protected EAP (PEAP) to provide a common protocol for differing authentication mechanisms. EAP enables the dynamic selection of the authentication mechanism at authentication time based on information transmitted in the Access-Request. Prime Access Registrar provides the following EAP services:
- EAP-FAST
- EAP-GTC
- EAP-LEAP
- EAP-MD5
- EAP-MSChapV2
- EAP-Negotiate
- EAP-SIM
- EAP-Transport Level Security (TLS)
- EAP-Tunneled TLS (TTLS)
- PEAP Version 0 (Microsoft PEAP)
- PEAP Version 1 (Cisco PEAP)

See Chapter 9, “Extensible Authentication Protocols,” for detailed information about properties used in EAP-type services.

File

Specify the file service when you want Cisco Prime Access Registrar’s RADIUS Server to perform local accounting using a specific file. Every file Service in your configuration will cause a file with the configured name to be created when the server is started, even if the service is not being invoked by any request packets. Table 4-12 lists the properties used for a file service.

Table 4-12 File Service Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required; must be set to group for a group service.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Name of script to run when the service starts.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Name of script to run when the service ends.</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; the default is DropPacket. This property defines how Cisco Prime Access Registrar handles requests if all servers listed in the RemoteServers properties are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: AcceptAll, DropPacket, or RejectAll.</td>
</tr>
<tr>
<td>OutageScript</td>
<td>Optional; if you set this property to the name of a script, Cisco Prime Access Registrar runs it when an outage occurs. This property allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
<tr>
<td>FilenamePrefix</td>
<td>Required; a string that specifies where Cisco Prime Access Registrar writes the account records. It must be either a relative or absolute path. When you specify a relative path, it must be relative to the $INSTALL/logs directory. When you specify an absolute path, the server must be able to reach it. The default is Accounting.</td>
</tr>
<tr>
<td>MaxFileSize</td>
<td>Optional; stored as a string, but is composed of two parts, a number and a units indicator (&lt;n&gt; &lt;units&gt;) in which the unit is one of: K, Kilobyte, Kilobytes, M, Megabyte, Megabytes, G, Gigabyte, Gigabytes. The default is ten megabytes.</td>
</tr>
</tbody>
</table>
Cisco Prime Access Registrar opens the file when it starts the RADIUS server and closes the file when you stop the server. Prime Access Registrar flushes the accounting record to disk before it acknowledges the request.

Based on the maximum file size and age you have specified, Prime Access Registrar closes the accounting file, moves it to a new name, and reopens the file as a new file. The name Prime Access Registrar gives this accounting file depends on its creation and modification dates.

- If the file was created and modified on the same date, the filename is `FileNamePrefix-<yyyymmdd><n>.log`. The date is displayed as year, month, day, number.
- If the file was created on one day and modified on another, the filename is `FileNamePrefix-<yyyymmdd>-<yyyymmdd><n>.log`. The dates are creation, modification, and number.

### Group

A group service contains a list of references to other services and specifies whether the responses from each of the services should be handled as a logical AND or a logical OR function. You specify AND or OR in the Result-Rule attribute of Group Services. The default value is AND.

Table 4-13 lists the properties used to configure a **group** service.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required; must set it to group.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; name of script to run when the service starts.</td>
</tr>
</tbody>
</table>
If Result-Rule is set to AND, the response from the Group Service is positive if each of the services referenced return a positive result. The response is negative if any of the services reference return a negative result. If Result-Rule is set to OR, the response from the Group Service is positive if any of the services referenced return a positive result. The response is negative if all the referenced services return a negative result.

When the Result-Rule attribute is set to AND or OR, each referenced service is accessed sequentially, and the Group Service waits for a response from the first referenced service before moving on to the next service (if necessary). If a service takes a long time to respond, that causes a delay in sending the request to the next referenced server.

The ResultRule settings parallel-and and parallel-or are similar to AND and OR settings, except that they ask each referenced service to process the request simultaneously instead of asking each referenced server sequentially, thereby saving processing time.

A parallel-and setting might respond with its own reply as soon as it receives a negative response, but otherwise must wait for all responses before it can respond with a positive reply. Likewise, a parallel-or might respond as soon as it receives a positive response, but otherwise must wait for all responses before it can reply with a negative response.

If a service referenced from a Group Service is of type RADIUS and if Accounting-Requests are being processed by the Group Service, setting the AckAccounting property in the remote server will affect the behavior of the parallel-or Group Service. This is because if AckAccounting is set to FALSE, the RADIUS Remote Server will not wait for the response from the remote server but returns a response immediately. Since the Group Service is set to parallel-or, after it receives the response from the RADIUS service, it is free to send a response itself. This will have the effect that a response is sent very quickly from the Group Service acknowledging the Accounting-Request and responses from the other referenced services are handled as the arrive.

Note that since AckAccounting was set to FALSE, there is no guarantee that the Remote Server successfully processed the request. Since it is a RADIUS Remote Server, the Prime Access Registrar server attempts for MaxTries to send the request to the server and to get back an acknowledgment, but if that fails, there will be no indication to the client about that event. The acknowledgment to the client has been sent long before.
Java

Specify the **java** service type when you want to create a custom service and use a script for authentication, authorization, or accounting. Table 4-14 lists the properties required to configure a java service.

A java service uses an extension point script to provide the service’s functionality and handles both RADIUS and TACACS requests for authentication, authorization, and accounting.

**Table 4-14 Java Service Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required; must set it to java.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Name of script to run when the service starts.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Name of script to run when the service ends.</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; the default is <strong>DropPacket</strong>. This property defines how Cisco Prime Access Registrar handles requests if all servers listed in the <strong>RemoteServers</strong> properties are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: <strong>AcceptAll</strong>, <strong>DropPacket</strong>, or <strong>RejectAll</strong>.</td>
</tr>
<tr>
<td>OutageScript</td>
<td>Optional; if you set this property to the name of a script, Cisco Prime Access Registrar runs it when an outage occurs. This property allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
<tr>
<td>ClassName</td>
<td>Set to the name of a class that implements the Extension interface.</td>
</tr>
<tr>
<td>InitializeArg</td>
<td>Optional; set to a string to be passed to the Initialize method if the class implements the optional ExtensionWithInitialization interface.</td>
</tr>
</tbody>
</table>

LDAP

Specify the **ldap** service type when you want to use a particular LDAP remote server for authentication and/or authorization. Table 4-15 lists the properties used to configure an LDAP service.

When using LDAP for authentication and a local database for authorization, ensure that the usernames in both locations are identical with regard to case sensitivity.

**Table 4-15 LDAP Service Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required, must set it to ldap</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Name of script to run when the service starts.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Name of script to run when the service ends.</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; the default is <strong>DropPacket</strong>. This property defines how Cisco Prime Access Registrar handles requests if all servers listed in the <strong>RemoteServers</strong> properties are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: <strong>AcceptAll</strong>, <strong>DropPacket</strong>, or <strong>RejectAll</strong>.</td>
</tr>
</tbody>
</table>
Local

Specify local when you want the Cisco Prime Access Registrar server to perform the authentication and authorization using a specific UserList. For more information, see the “UserLists” section on page 4-3. Table 4-16 lists the properties used to configure a local service.

### Table 4-16  Local Service Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required, must set it to <strong>local</strong>.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; name of script to run when the service starts.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; name of script to run when the service ends.</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; the default is <strong>DropPacket</strong>. This property defines how Cisco Prime Access Registrar handles requests if all servers listed in the <strong>RemoteServers</strong> properties are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: <strong>AcceptAll</strong>, <strong>DropPacket</strong>, or <strong>RejectAll</strong>.</td>
</tr>
<tr>
<td>OutageScript</td>
<td>Optional; if you set this property to the name of a script, Cisco Prime Access Registrar runs it when an outage occurs. This property allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
<tr>
<td>UserLists</td>
<td>Required; this object contains all of the individual UserLists, which in turn, contain the specific users stored within Prime Access Registrar. Cisco Prime Access Registrar references each specific UserList by <strong>name</strong> from a Service whose type is set to <strong>local</strong>. When Cisco Prime Access Registrar receives a request, it directs it to a Service. When the Service has its type property set to <strong>local</strong>, the Service looks up the user’s entry in the specific UserList and authenticates and/or authorizes the user against that entry.</td>
</tr>
</tbody>
</table>
ODBC

Specify **odbc** when you want to use an ODBC service for authentication, authorization and accounting through an ODBC data store. Use an ODBC service to authenticate and authorize an access requests by querying user information through ODBC and to insert accounting records into a data store through ODBC. **Table 4-17** lists the properties used to configure an ODBC service.

**Table 4-17**  ODBC Service Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required; must set it to <strong>odbc</strong>.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; name of script to run when the service starts.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; name of script to run when the service ends.</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; the default is <strong>DropPacket</strong>. This property defines how Cisco Prime Access Registrar handles requests if all servers listed in the <strong>RemoteServers</strong> properties are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: <strong>AcceptAll</strong>, <strong>DropPacket</strong>, or <strong>RejectAll</strong>.</td>
</tr>
<tr>
<td>OutageScript</td>
<td>Optional; if you set this property to the name of a script, Cisco Prime Access Registrar runs it when an outage occurs. This property allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
<tr>
<td>MultipleServersPolicy</td>
<td>Required; must be set to either <strong>Failover</strong> or <strong>RoundRobin</strong>.</td>
</tr>
<tr>
<td></td>
<td>When you set it to <strong>Failover</strong>, Cisco Prime Access Registrar directs requests to the first server in the list until it determines the server is offline. At which time, Cisco Prime Access Registrar redirects all requests to the next server in the list until it finds a server that is online.</td>
</tr>
<tr>
<td></td>
<td>When you set it to <strong>RoundRobin</strong>, Cisco Prime Access Registrar directs each request to the next server in the RemoteServers list to share the resource load across all of the servers listed in the RemoteServers list.</td>
</tr>
<tr>
<td>RemoteServers</td>
<td>Required; an indexed list from 1 to <strong>&lt;n&gt;</strong>. Each entry in the list is the name of a RemoteServer.</td>
</tr>
</tbody>
</table>

**ODBC-Accounting**


**Prepaid Services**

RADIUS

Specify the **radius** service type when you want to use a particular RADIUS remote server for authentication and authorization. Table 4-18 lists the properties used to configure a RADIUS service.

**Table 4-18  RADIUS Service Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required; must set it to <strong>radius</strong>.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; name of script to run when the service starts.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; name of script to run when the service ends.</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; the default is <strong>DropPacket</strong>. This property defines how Cisco Prime Access Registrar handles requests if all servers listed in the <strong>RemoteServers</strong> properties are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: <strong>AcceptAll</strong>, <strong>DropPacket</strong>, or <strong>RejectAll</strong>.</td>
</tr>
<tr>
<td>OutageScript</td>
<td>Optional; if you set this property to the name of a script, Cisco Prime Access Registrar runs it when an outage occurs. This property allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
<tr>
<td>MultipleServersPolicy</td>
<td>Required; must be set to either <strong>Failover</strong> or <strong>RoundRobin</strong>. When you set it to <strong>Failover</strong>, Cisco Prime Access Registrar directs requests to the first server in the list until it determines the server is offline. At which time, Cisco Prime Access Registrar redirects all requests to the next server in the list until it finds a server that is online. When you set it to <strong>RoundRobin</strong>, Cisco Prime Access Registrar directs each request to the next server in the RemoteServers list to share the resource load across all of the servers listed in the RemoteServers list.</td>
</tr>
<tr>
<td>RemoteServers</td>
<td>Required; an indexed list from 1 to <strong>&lt;n&gt;</strong>. Each entry in the list is the name of a RemoteServer.</td>
</tr>
</tbody>
</table>

Radius Query

Prime Access Registrar supports a new service type called radius-query that can be used to query cached data through RADIUS packets. This radius-query service contains a list of session managers to be queried from and a list of (cached) attributes to be returned in the Access-Accept packet in response to a RADIUS Query request. Prime Access Registrar also supports caching and querying of multivalued attributes.

The RADIUS Query service should be selected through an extension point script or through the Rule and Policy Engine by setting it to a new environment variable named Query-Service. The reason for this is that the RADIUS Query request comes in as an Access-Request and the server has no way of knowing whether it is a RADIUS Query request or normal authentication request. Setting the Query-Service environment variable tells the Prime Access Registrar server that the request is a RADIUS Query request so the Prime Access Registrar server can process the request with the radius-query service set in the Query-Service environment variable.

When a RADIUS Query service is selected to process an Access-Request, it queries the configured list of Session Managers for a matching record using the QueryKey value configured in the session-cache Resource Manager referenced under these Session Managers as key. If a matching record is found, an
Access-Accept containing a list of cached attributes present (based on the configuration) in the matched record is sent back to the client. If the session cache contains a multivalued attribute, all values of that attribute are returned in the response as a multivalued attribute. If there is no matching record, an Access-Reject packet is sent to the client.

Prime Access Registrar introduces scripting points at the Session Manager level along with automated programmable interfaces (APIs) to access cached information present in the session record. You can use these scripting points and APIs to write extension point scrips to modify the cached information.

The following example shows the default configuration of a radius-query service:

```
[ //localhost/Radius/Services/radius-query ]
Name = radius-query
Description =
Type = radius-query
IncomingScript =
OutgoingScript =
SessionManagersToBeQueried/
AttributesToBeReturned/
```

Table 4-19 lists the properties used to configure a RADIUS Query service.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required; must set it to <code>radius query</code>.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; name of script to run before this service starts processing on the request.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; name of script to run after this service completes processing on the request.</td>
</tr>
<tr>
<td>SessionManagersToBeQueried</td>
<td>Lists Session Managers to be queried for the target record. If this list is empty, all Session Managers having session-cache Resource Managers will be queried for the target record. Otherwise, only those SessionManagers configured under <code>SessionManagerToBeQueried</code> are queried. If the targeted record is found in a Session Manager, the query stops and the response is returned to the client.</td>
</tr>
<tr>
<td>AttributesToBeReturned</td>
<td>Lists attributes to be returned if present in a matched record. If this list is empty, all attributes cached in a matched session are returned. If a configured attribute is not present in the matched record, that attribute is ignored. <strong>Note</strong> The User-Password attribute will not be returned in query responses and cannot be configured under <code>AttributesToBeReturned</code>.</td>
</tr>
</tbody>
</table>

When an Access-Request packet is received by the Prime Access Registrar server, the session-cache Resource Manager caches the configured attributes in the session with the configured QueryKey as the key to the cached data. In the TAL solution, the QueryKey will usually be Framed-IP-Address. If an Accounting-Requestor Accounting-Start packet is received for the same session, the cached data is updated if necessary. If there is a multivalued attribute in the Access-Request packet or Accounting-Request packet, the Prime Access Registrar server caches all the values of that attributes.
In TAL, when the SSG receives an IP packet originating from a user unknown to the SSG, it sends an Access-Request packet to the Prime Access Registrar server in which the User-Name and Framed-IP-Address attributes both contain the user’s source IP address, and the Service-Type is set to Outbound, among other attributes. These attributes and their values distinguish RADIUS Query requests from normal authentication requests in TAL.

**Note**

In solutions other than TAL, the criterion that distinguishes RADIUS Query requests from normal authentication requests might be different.

A new environment variable, Query-Service, can be set to the name of a radius-query service, in an extension point script, or through the Rule and Policy engine so the Prime Access Registrar server knows the current request is a RADIUS Query request and processes it with the radius-query service value set in the Query-Service environment variable.

### API Calls

Prime Access Registrar provides several new API calls you can use to get, put, and delete the cached attributes present in the session record.

The entry point function changes slightly to take a fifth argument which is a pointer to a structure containing the new API calls:

```c
typedef int (REXAPI * RexEntryPointFunction)(
    int iScriptingPoint,
    rex_AttributeDictionary_t* pRequest,
    rex_AttributeDictionary_t* pResponse,
    rex_EnvironmentDictionary_t* pRadius,
    rex_SessionRecord_t* pSession);
```

However, you can continue to write extension point scripts with four arguments as well, for example without the pSession argument.

The following are API calls and their functionality. All these API calls fail gracefully when they are invoked from any scripting point other than the Session Manager scripting points.

**const char* get**

```c
const char* get(
    rex_SessionRecord_t* pSession,
    const char* pszAttribute,
    int <iIndex>,
    abool_t* <pbMore>
);
```

This API returns the value of the <iIndex>'d instance of the attribute cached in the session, represented as a string. When the session does not contain the attribute, an empty string is returned. When <pbMore> is non-zero, this method sets <pbMore> to TRUE when more instances of the same attribute exist after the one returned and to FALSE otherwise. This can be used to determine whether another call to get() method should be made to retrieve other instances of the same attribute.

**abool_t put**

```c
abool_t put(
    rex_SessionRecord_t* pSession,
    const char* pszAttribute,
    const char* <pszValue>,
    int <iIndex>
);
```
When `<iIndex>` equals the special value REX_REPLACE, this method replaces any existing instances of `<pszAttribute>` with a single value in the session. When `<iIndex>` equals the special value REX_APPEND, it appends a new instance of `<pszAttribute>` to the end of the list of existing instances of `<pszAttribute>`. When `<iIndex>` equals the special value REX_AUGMENT, this method only puts `<pszAttribute>` when it does not already exist. Otherwise, a new instance of `<pszAttribute>` is inserted/replaced at the position indicated. This method returns TRUE if it is able to cache the attribute successfully and FALSE otherwise.

```c
abool_t remove
abool_t remove(
    rex_SessionRecord_t* pSession,
    const char* pszAttribute,
    int <iIndex>
);
```

This method removes the `<pszAttribute>` from the session. When `<iIndex>` equals the special value REX_REMOVE_ALL, this method removes any existing instances of `<pszAttribute>`. Otherwise, it removes the instance of `<pszAttribute>` at the position indicated. It returns FALSE when `<pszAttribute>` is not present at any index in the session record and returns TRUE otherwise.

```c
rex_SessionInfo_t* getSessionInfo(rex_SessionRecord_t* pSession )
```

This method returns the pointer to a structure that contains the other session-related information, like Session Id, Session Start time, Session Last Accessed Time, present in the session record. The structure that holds this information will appear as follows:

```c
typedef struct rex_SessionInfo_s
{
    auint32_t iSessionId;
    auint32_t tSessionStartTime;
    auint32_t tSessionLastAccessedTime;
} rex_SessionInfo_t;
```

### Tcl API calls

To use the extension point scripts written in Tcl, define the procedure at the session manager level as shown below:

```tcl
proc test { request response environ session } {{
}
```

There is a fourth argument `session` that needs to be passed to the Tcl procedure and the API calls that are intended to operate on the session record need to use this `session` dictionary.

API calls in Tcl have the same meaning with same number arguments and return values as described in Rex. The only difference is that the API `getSessionInfo` will not return a structure as in Rex but it will return the info as a string, as in the following example:

```
Session-ID=1, Session-Start-Time=1102099334, Session-Last-Accessed-Time=1102099334
```

### Java API calls

There are two new interfaces `ExtensionForSession` and `ExtensionForSessionWithInitialization` and the customers wishing to use the extension point scripts written in Java at the session manager level needs to implement one of these interfaces.

The `runExtension` method of these interfaces will look as below:

```java
public int runExtension
( int iExtensionPoint,
```
AttributeDictionary request,
AttributeDictionary response,
EnvironmentDictionary environment,
SessionRecord session
);

API calls that are intended to operate on session record needs to use this ‘session’ dictionary.

API calls in Java have the same meaning with same number arguments and return values as described in Rex. The only difference is that the API getSessionInfo will not return a structure as in Rex but it will return the info as a string. For example:

    Session-ID=1, Session-Start-Time=1102099334, Session-Last-Accessed-Time=1102099334

Existing scripts written in any of these three languages will not be affected with the introduction of the new ‘session dictionary’ argument. And the customers can use a script with any number of arguments (i.e with or without the last ‘session dictionary’ argument) at any extension point script. If there is no session to operate on, for example when the customer is trying to use session dictionary argument at an extension point other than session manager’s, the Prime Access Registrar gracefully returns an error logging the appropriate message.

The simple replace or add if it does not exist model can still be used for simple modifications as before without the need to write a script. If the cached attributes are updated in the IncomingScript and if customers do not want them to be touched or updated again when the processing reaches session-cache resource manager, they can set the OverwriteAttributes property of the session-cache resource manager to FALSE so that the session-cache resource manager will not operate on this packet.

RADIUS-Session

A new Service step has been added in the processing of Access-Request and Accounting packets. This is an additional step after the AA processing for Access packet or Accounting processing for Accounting packet, but before the local session management processing. The Session-Service should have a service type of radius-session.

An environment variable Session-Service is introduced to determine the Session-Service dynamically. You can use a script or the rule engine to set the Session-Service environment variable. See Cross Server Session and Resource Management, page 1-9 for more information on RADIUS-Session.

Rex

Specify the rex service type when you want to create a custom service and use a script for authentication, authorization, or accounting. Table 4-20 lists the properties required to configure a rex service.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required; must be set to rex.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; name of script to run when the service starts.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; name of script to run when the service ends.</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; the default is DropPacket. This property defines how Cisco Prime Access Registrar handles requests if all servers listed in the RemoteServers properties are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: AcceptAll, DropPacket, or RejectAll.</td>
</tr>
</tbody>
</table>
WiMAX

Prime Access Registrar uses the Extensible Authentication Protocol (EAP) to enable the WiMAX feature. It also caches the IP attributes and Mobility Keys that are generated during network access authentication. To enable caching of the WiMAX attributes, you must configure the respective resource managers. See WiMAX in Cisco Prime Access Registrar, page 10-2, for more information on WiMAX.

Diameter

Diameter works with the rule policy engine to perform the routing for multiple peers. The following are the multiple peer policies supported with the proxy service to route the traffic:

- RoundRobin
- FailOver
- IMSI Range Based

The following configuration is used to add Diameter proxy without Sticky session configuration:

```
[ //localhost/Radius/Services/dia-proxy ]
Name = dia-proxy
Description =
Type = diameter
IncomingScript~ =
OutgoingScript~ =
EnableSticky = FALSE
MultiplePeersPolicy = RoundRobin
PeerTimeOutPolicy = SendError
```
DiaRemoteServers/
   Entries 1 to 2 from 2 total entries
   Current filter: <all>

   proxy_rm/
      Name = proxy_rm
      Metric = 1
      Weight = 0
      IsActive = TRUE
   proxy_rm1/
      Name = proxy_rm1
      Metric = 2
      Weight = 0
      IsActive = TRUE

The following configuration is used to add Diameter proxy with Sticky session configuration:

[ //localhost/Radius/Services/dia-proxy ]
Name = dia-proxy
Description =
   Type = diameter
   IncomingScript~ =
   OutgoingScript~ =
   EnableSticky = TRUE
   StickySessionKey = User-Name#1
   StickyCreationCmdList = 265
   StickyDeletionCmdList = 275
   MultiplePeersPolicy = RoundRobin
   PeerTimeOutPolicy = SendError
DiaRemoteServers/
   Entries 1 to 2 from 2 total entries
   Current filter: <all>

   proxy_rm/
      Name = proxy_rm
      Metric = 1
      Weight = 0
      IsActive = TRUE
   proxy_rm1/
      Name = proxy_rm1
      Metric = 2
      Weight = 0
      IsActive = TRUE

The following configuration is used to add Diameter proxy with IMSI range based load balancing configuration:

[ //localhost/Radius/Services/dia-proxy ]
Name = dia-proxy
Description =
   Type = diameter
   IncomingScript~ =
   OutgoingScript~ =
   EnableSticky = TRUE
   StickySessionKey = User-Name#1
   StickyCreationCmdList = 265
   StickyDeletionCmdList = 275
   MultiplePeersPolicy = IMISRANGEBased
   PeerTimeOutPolicy = SendError
IMISRanges/
   Entries 1 to 1 from 1 total entries
   Current filter: <all>

   112156000000001-112156001000000/
The following configuration is used to add Diameter local:

```ini
[ //localhost/Radius/Services/dia-local ]
Name = dia-local
Description =
Type = diameter
Realm = abc.com
Role = Local
IncomingScript~ =
OutgoingScript~ =
AuthenticationService = local-users
AccountingService = local-file
DiaRemoteServers/
   Entries 1 to 1 from 1 total entries
   Current filter: <all>
   murdcoh-ar1/
      Name = murdcoh-ar1
      HostName = murdoch-ar1
      IsVendorSpecific = FALSE
      ApplicationID = 1
      Metric = 2
```

The following configuration is used to add Diameter relay:

```ini
[ //localhost/Radius/Services/dia-relay ]
Name = dia-relay
Description =
Type = diameter
Realm = abc.com
Role = relay
DiaRemoteServers/
   Entries 1 to 1 from 1 total entries
   Current filter: <all>
   hss1/
      Name = hss1
      HostName = 10.77.240.69
      IsVendorSpecific = FALSE
      ApplicationID = 1
      Metric = 2
```

The following configuration is used to add Diameter redirect:

```ini
[ //localhost/Radius/Services/dia-redirect ]
```
Name = dia-relay
Description =
Type = diameter
Realm = abc.com
Role = redirect
DiaRemoteServers/
    Entries 1 to 1 from 1 total entries
    Current filter: <all>

redirectserver/
    Name = redirectserver
    HostName = 10.77.240.69
    IsVendorSpecific = FALSE
    ApplicationID = 1
    Metric = 2
Table 4-21 describes the Diameter Service properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; name of the Diameter server.</td>
</tr>
<tr>
<td>Realm</td>
<td>Required; realm of the route. Must be unique for a route table.</td>
</tr>
<tr>
<td>Incoming Script</td>
<td>Optional; enabled when role is set to Proxy or Local. When set, must be the name of a known incoming script. Prime Access Registrar runs the IncomingScript before proxying the diameter packet to the remote diameter server.</td>
</tr>
<tr>
<td>Outgoing Script</td>
<td>Optional; enabled when role is set to Proxy or Local. When set, must be the name of a known outgoing script. Prime Access Registrar runs the OutgoingScript after it receives the response from the remote Diameter server.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the Diameter server.</td>
</tr>
<tr>
<td>Role</td>
<td>Required; specifies the role that the diameter entity will play in resolving messages matching the realm.</td>
</tr>
<tr>
<td></td>
<td>The role can be any one of the following:</td>
</tr>
<tr>
<td></td>
<td>Relay - Application acting as a Relay Agent.</td>
</tr>
<tr>
<td></td>
<td>Redirect - Application acting as a Redirect Agent.</td>
</tr>
<tr>
<td></td>
<td>Proxy - Application acting as a Proxy Agent. When the role is set to Proxy, the IncomingScript and OutgoingScript points are enabled.</td>
</tr>
<tr>
<td></td>
<td>Local - Application processes the requests locally. When the role is set to Local, the AuthenticationService and AccountingService are enabled.</td>
</tr>
<tr>
<td></td>
<td>By default, the Proxy option is selected. However, you can select another option from the drop-down list.</td>
</tr>
<tr>
<td>AuthenticationService</td>
<td>Required; used when service is configured to process the diameter requests locally. Set to valid service of type (local/ldap/odbc) to authenticate the user. This field is displayed when you select the role type as ‘Local’ in the Role field.</td>
</tr>
<tr>
<td>AccountingService</td>
<td>Required; used when service is configured to process the accounting requests locally. Set to valid accounting service of type (file/odbc-accounting) to write the accounting records. This field is displayed when you select the role type as ‘Local’ in the Role field.</td>
</tr>
<tr>
<td>Type</td>
<td>Required; specifies the service type. The service type ‘Diameter’ is automatically displayed in this field.</td>
</tr>
</tbody>
</table>

**Peer Definitions Tab**

This tab is displayed when you select the ‘Local’, ‘Relay’, or ‘Redirect’ option in the Role field.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; name of the peer.</td>
</tr>
<tr>
<td>Host Name</td>
<td>Required; the hostname or IP address of the peer. The hostname must exist in the client list for the route to be active.</td>
</tr>
<tr>
<td>Metric</td>
<td>Required; metric value for the peer entry. The higher the value the lower the preference. The highest value of preference is 0.</td>
</tr>
</tbody>
</table>
Table 4-21   Diameter Service Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VendorSpecific</td>
<td>Required; the default is FALSE. If set to FALSE, the application is ordinary application and user is prompted to enter the ApplicationID. If set to TRUE, the application is a VendorSpecific Application. User is prompted to enter VendorSpecificApplicationID and VendorID.</td>
</tr>
<tr>
<td>VendorID</td>
<td>Required; specifies the VendorID for the application.</td>
</tr>
<tr>
<td>Example:</td>
<td>DIAMETER 3GPP Cx APPLICATION</td>
</tr>
<tr>
<td></td>
<td>VendorSpecificApplicationID 16777216</td>
</tr>
<tr>
<td></td>
<td>VendorID</td>
</tr>
<tr>
<td>VendorSpecificApplicationID</td>
<td>Required; specifies the integer value for the vendor specific application.</td>
</tr>
<tr>
<td>ApplicationID</td>
<td>Required; application used in the route. The application Id should be available in /Advanced/Diameter/Applications.</td>
</tr>
</tbody>
</table>

**Applications tab**

This tab is displayed when you select the ‘Proxy’ option in the Role field.

<table>
<thead>
<tr>
<th>Name</th>
<th>Required; name of the application.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>The description of the application.</td>
</tr>
<tr>
<td>ApplicationID</td>
<td>Required; specifies the unique integer value for the application. It represents the applicationid of the Application used for load balancing the diameter messages.</td>
</tr>
<tr>
<td>EnableSticky</td>
<td>Required; default is FALSE. If set to True, the sticky entries for load balancing is enabled and the user is prompted to enter the values for StickySessionKey, StickyCreationCmdList, and StickyDeletionCmdList.</td>
</tr>
</tbody>
</table>
| DeMultiplexCCTerminateRequest| Optional; default is FALSE. If set to True, Prime Access Registrar generates and sends multiple Credit Control Update (CCR-U) requests corresponding to an incoming diameter Credit Control Termination (CCR-T) request, while proxying Gy messages between the Gateway GPRS Support Node (GGSN) and Online charging system (OCS). 

The CCR-U requests are generated based on the number of RGs present in CCR-T message. |
| MultiplePeersPolicy         | Required; must be set to RoundRobin, FailOver, or IMSIRangeBased. Policy used by the Prime Access Registrar server to load balance the peers. |
StickySessionKey

Required; used as the sticky key for mapping the sticky sessions. Set the value to a valid AVP in order to use the sticky key for maintaining diameter sessions. This ensures that Prime Access Registrar maps the request to the same server for all the subsequent messages using the sticky key. For example, set StickyAVP “Session-Id”.

When the Prime Access Registrar server receives the CCR-I request, Prime Access Registrar extracts the Session-Id from the request packet, maps the Session to the peer configured in the list, and forwards the request to the chosen peer. Prime Access Registrar chooses the same peer for all the subsequent messages (CCR-Update/CCR-Terminate) with same Session-Id.

StickyCreationCmdList

Required; specifies the command list to create the sticky entries. Specify the list of ‘||’ separated command code, AVP name, and its value to create the sticky sessions.

The following is the StickyCreationCmdList format:

```
<commandcode1>::<AVPName1=Value1> ||
<commandcode2>::<AVPName2=Value2> ||<commandcode3>
```

For example, if the sticky session entries need to created based on command code ‘265’ or based on command code ‘271’ with Accounting-Record-Type value as 2, use the format below:

Set StickyCreationCmdList “265||271:: Accounting-Record-Type=2”

StickyDeletionCmdList

Required; specifies the command list to delete the sticky entries. Specify the list of ‘||’ separated command code, AVP name, and its value to delete the sticky sessions.

The following is the StickyDeletionCmdList format:

```
<commandcode1>::<AVPName1=Value1> ||
<commandcode2>::<AVPName2=Value2> ||<commandcode3>
```

For example, if the sticky session entries need to deleted based on command code ‘271’ with Accounting-Record-Type value as 4, use the format below:

Set StickyDeletionCmdList “271:: Accounting-Record-Type=4”

Peer Definitions Proxy tab

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; name of the peer.</td>
</tr>
<tr>
<td>Host Name</td>
<td>Required; hostname or IP address of the peer. The HostName must exist in the client list for the route to be active.</td>
</tr>
<tr>
<td>Metric</td>
<td>Required; metric value for this peer entry. The higher the value the lower the preference. The highest value of preference is 0.</td>
</tr>
</tbody>
</table>
Table 4-21  Diameter Service Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>Required; default value is 0. Specifies the weight percentage for which the service needs to load balance the peer.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> When you set the weight to a value other than 0, the weight should be in multiples of 10 and the sum of the weights configured in the peer list should be equal to 100.</td>
</tr>
<tr>
<td>IMSIRanges</td>
<td>Required; used for load balancing. The value is set to comma separated values of IMSI Ranges.</td>
</tr>
<tr>
<td></td>
<td>For example, set IMSIRanges “112156000000001-112156001000000,112156010000001-11215601100000”</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> Prime Access Registrar uses the AVP configured in StickyAVP property to check whether the IMSI is in valid range.</td>
</tr>
</tbody>
</table>

### M3UA

Prime Access Registrar supports the M3UA service, which is used to fetch MSISDN from IMSI through RADIUS Packets. The M3UA service sends a SendRoutingInfoForLCS(SRIForLCS) request that contains the IMSI information to the remote HLR. The HLR sends the MSISDN in response. To fetch the MSISDN information from IMSI, you need to configure the SIGTRAN-M3UA remote server where Prime Access Registrar is installed, see Configuring M3UA Service for more information.

The M3UA service checks for IMSI environment variable to fetch the MSISDN information. If there is no IMSI environment variable set, then the User-Name in the Radius Access-Request is used as IMSI to fetch the MSISDN information. The fetched MSISDN is copied to the AuthorizationInfo environment variable where you can write a script to copy the environment variable to any attribute of your choice.

**Note** M3UA service supports fetching the MSISDN only through SIGTRAN-M3UA interface.

The following shows an example configuration of M3UA service:

```plaintext
[ //localhost/Radius/Services/FetchMSISDN ]
Name = FetchMSISDN
Description =
Type = m3ua
IncomingScript- =
OutgoingScript- =
OutageScript- =
OutagePolicy- = RejectAll
RemoteServers/
```

### Table 4-22 M3UA Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required; must set to M3UA service.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; when set, must be the name of a known incoming script. Cisco Prime Access Registrar runs the IncomingScript after it receives the response.</td>
</tr>
</tbody>
</table>
Session Managers

You can use Session Managers to track user sessions. The Session Managers monitor the flow of requests from each NAS and detect the session state. When requests come through to the Session Manager, it creates sessions, allocates resources from appropriate Resource Managers, and frees and deletes sessions when users log out.

The Session Manager enables you to allocate dynamic resources to users for the lifetime of their session. You can define one or more Session Managers and have each one manage the sessions for a particular group or company.

**Table 4-22 M3UA Properties (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OutgoingScript</td>
<td>Optional; when set, must be the name of a known outgoing script. Cisco Prime Access Registrar runs the OutgoingScript just before it sends the proxy request to the remote server.</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; the default is DropPacket. This property defines how Prime Access Registrar handles requests if all servers listed in the RemoteServers properties are unavailable (that is, all remote RADIUS servers are not available). You must set it to one of the following: AcceptAll, DropPacket, or RejectAll.</td>
</tr>
<tr>
<td>OutageScript</td>
<td>Optional; set this property to the name of a script. Prime Access Registrar runs the script when an outage occurs. It allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
<tr>
<td>RemoteServers</td>
<td>Required; an indexed list from 1 to &lt;n&gt;. Each entry in the list is the name of a RemoteServer of type SIGTRAN-M3UA.</td>
</tr>
</tbody>
</table>

**Note**

Session record size is limited by the operating system (OS) paging size (4 KB in Linux). If a request triggers creation of a session that exceeds the OS paging size, the request will be dropped and the session will not be created.

**Note**

In this release of Prime Access Registrar, the memory capacity is enhanced to store more than 4 million active sessions by storing the active session records in database server instead of storing it in the main memory. The capacity is dependent on the number of attributes that are being captured for each session.

**Note**

If the disk partition where Prime Access Registrar stores session backing store data (usually the disk partition where Prime Access Registrar is installed, such as /opt/CSCOar) is full, the subsequent packets that try to create sessions will be dropped and no sessions will be created due to lack of disk space.

Session Managers use Resource Managers, which in turn, manage a pool of resources of a particular type. Table 4-23 lists the Session Manager properties.
Prime Access Registrar adds IncomingScript, OutGoingScript, and SessionKey properties. The IncomingScript is run as soon as the session is acquired. The OutGoingScript is run just before the session is written to backing store. The SessionKey property sets the session key value for the Session Manager.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Session Managers list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the Session Manager.</td>
</tr>
<tr>
<td>Type</td>
<td>Set to local or remote. Local is the traditional session manager that</td>
</tr>
<tr>
<td></td>
<td>maintains sessions in memory and has good performance. The remote</td>
</tr>
<tr>
<td></td>
<td>session manager operates on a remote ODBC database, and its</td>
</tr>
<tr>
<td></td>
<td>performance is highly dependent on the performance of the ODBC database.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; name of script to run when the service starts. This script is run</td>
</tr>
<tr>
<td></td>
<td>as soon as the session is acquired in Prime Access Registrar.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; script to be run just before the session is written to backing</td>
</tr>
<tr>
<td></td>
<td>store.</td>
</tr>
<tr>
<td>SessionTimeOut</td>
<td>The SessionTimeOut property is optional; no value for this property</td>
</tr>
<tr>
<td></td>
<td>means the session timeout feature is disabled.</td>
</tr>
<tr>
<td></td>
<td>Used in conjunction with /Radius/Advanced/SessionPurgeInterval for the</td>
</tr>
<tr>
<td></td>
<td>session timeout feature. Enables the session timeout feature for a</td>
</tr>
<tr>
<td></td>
<td>Session Manager. If the SessionTimeOut property is set to a value under</td>
</tr>
<tr>
<td></td>
<td>a session manager, all sessions that belong to that session manager will</td>
</tr>
<tr>
<td></td>
<td>be checked for timeouts at each SessionPurgeInterval. If any sessions</td>
</tr>
<tr>
<td></td>
<td>have timed out, they will be released, and all resources associated with</td>
</tr>
<tr>
<td></td>
<td>those sessions are also released.</td>
</tr>
<tr>
<td></td>
<td>The SessionTimeOut property determines the timeout for a session. If the</td>
</tr>
<tr>
<td></td>
<td>time difference between the current time and the last update time is</td>
</tr>
<tr>
<td></td>
<td>greater than this property’s value, the session is considered to be stale.</td>
</tr>
<tr>
<td></td>
<td>The last update time of the session is the time at which the session was</td>
</tr>
<tr>
<td></td>
<td>created or updated.</td>
</tr>
<tr>
<td></td>
<td>The SessionTimeOut value is comprised of a number and a units indicator,</td>
</tr>
<tr>
<td></td>
<td>as in ( n \text{ units} ), where a unit is one of minutes, hours, days,</td>
</tr>
<tr>
<td></td>
<td>or weeks. The default unit is ‘days’.</td>
</tr>
<tr>
<td>AllowAccountingStartToCreateSession</td>
<td>Set to TRUE by default; start the session when the Prime Access Registrar server receives an Access Accept or an Accounting-Start. When set to FALSE, start the session when the Prime Access Registrar server receives an Access Accept.</td>
</tr>
<tr>
<td>Resource Managers</td>
<td>Ordered list of Resource Managers.</td>
</tr>
</tbody>
</table>
You can manage sessions with the two `aregcmd` session management commands: `query-sessions` and `release-sessions`. For more information about these two commands, see the `query-sessions`, page 2-9 and the `release-sessions`, page 2-9.

This section contains the following topics:

- **Session Creation**
Session Managers

- Session Notes
- Soft Group Session Limit
- Session Correlation Based on User-Defined Attributes

Session Creation

Cisco Prime Access Registrar Sessions can be created by two types of RADIUS packets:
- Access-Requests
- Accounting-Requests with an Acct-Status-Type attribute with a value of Start.

This allows Cisco Prime Access Registrar to monitor Sessions even when it is not allocating resources. For example, when Cisco Prime Access Registrar is being used as an “Accounting-Only” server (only receiving Accounting requests), it can create a Session for each Accounting “Start” packet it successfully processes. The corresponding Accounting “Stop” request will clean up the Session. Note, if a Session already exists for that NAS/NAS-Port/User (created by an Access-Request), Cisco Prime Access Registrar will not create a new one.

When you do not want Cisco Prime Access Registrar to create Sessions for Accounting “Start” requests, simply set the AllowAccountingStartToCreateSession property on the SessionManager to FALSE.

Session Notes

Session Notes are named text messages attached to a Session and are stored with the Session data, including resources allocated for a specific user session. This data, including Session Notes, can be retrieved and viewed using the aregcmd command query-sessions.

```bash
---> query-sessions /Radius/SessionManagers/session-mgr-2
sessions for /Radius/SessionManagers/session-mgr-2:
  S257 NAS: localhost, NAS-Port:1, User-Name: user1, Time: 00:00:08,
  IPX 0x1, GSL 1, USL 1, NOTES: "Date" "Today is 12/14/98.", "Requested
  IP Address" "1.2.3.4", "Framed-IP-Address" "11.21.31.4"
```

Session Notes can be created by Scripts using the Environment dictionary passed into each or by the Cisco Prime Access Registrar server. When more than one Session Note is added, the Session-Notes entry should be a comma-separated list of entry names.

Performing a TCL Script

To perform a TCL script:

**Step 1**
The Script should create an Environment dictionary entry using the Session Note name as the entry name, and the Session Note text as the entry value. For example:

```tcl
$environ put "Date" "Today is 12/15/08"
$environ put "Requested IP Address" "1.2.3.4"
```

**Step 2**
The Script should create or set an Environment dictionary entry with the name Session-Notes with a value that contains the name of the entries created. For example:

```tcl
$environ put "Session-Notes" "Date, Requested_IP_Address"
```
Performing a REX Script

To perform a REX script:

**Step 1**
The Script should create an Environment dictionary entry using the Session Note name as the entry name, and the Session Note text as the entry value. For example:

```plaintext
pEnviron->put(pEnviron, Date, "Today is 12/15/08.");
pEnviron->put(pEnviron, Request_IP_Address, "1.2.3.4");
```

**Step 2**
The Script should create/set an Environment dictionary entry with the name `Session-Notes` with a value that contains the name of the first entry created. For example:

```plaintext
pEnviron->put(pEnviron, "Session-Notes", "Date, Requested_IP_Address");
```

**Note**
Scripts creating Session Notes must be executed before the Session Management step takes place while processing a packet.

Cisco Prime Access Registrar will automatically create a Session Note if a packet is passed to a SessionManager and it already contains a `Framed-IP-Address` attribute in the packet’s Response dictionary. This IP address could come from a Profile, RemoteServer response, or from a previously executed script. For example, a Session output containing Session Notes when using the `aregcmd` command `query-session` would be as follows:

```plaintext
sessions for /Radius/SessionManagers/session-mgr-2:
  S257 NAS: localhost, NAS-Port:1, User-Name: user1, Time: 00:00:08,
  IPX 0x1, GSL 1, USL 1, NOTES: "Date" "Today is 12/14/08.", "Requested IP Address" "1.2.3.4", "Framed-IP-Address" "11.21.31.4"
```

Session Notes are also copied into the Environment dictionary after Session Management. The `Session-Notes` Environment dictionary entry will contain the names of all the Environment dictionary entries containing Session Notes.

In Prime Access Registrar, a major command is introduced—count-sessions. The `count-sessions /radius all` command helps to count the total sessions in Prime Access Registrar. The options are similar to the query-session command options. The query-session command displays cached attributes in addition to session details.

**Soft Group Session Limit**

Two new environment variables, `Group-Session-Limit` and `Current-Group-Count` (see rex.h), are set if the group session limit resource is allocated for a packet. These variables allow a script to see how close the group is to its session limit; one way to use this information is to implement a script-based soft limit. For example, you could use the Class attribute to mark sessions that have exceeded a soft limit of 80% -- as hard coded in the script (in a Tcl script called from `/Radius/OutgoingScript`):

```plaintext
set softlimit [ expr 0.8 * [ $environ get Group-Session-Limit ] ]
if { [ $environ get Current-Group-Count ] < $softlimit } {
    $response put Class 0
} else {
    $response put Class 1
}
```
Resource Managers

The soft limit itself is hard coded in the script; soft limits are not directly supported in the server. The action to be taken when the soft limit is exceeded (for example, Class = 1, and then the accounting software branches on the value of Class) is also the responsibility of the script and/or external software.

Session Correlation Based on User-Defined Attributes

All the session objects are maintained in one dictionary keyed by a string. You can define the keying material to the session dictionary through a newly introduced environment variable, Session-Key.

If the Session-Key is presented at the time of session manager process, it will be used as the key to the session object for this session. The Session-Key is of type string. By default, the Session-Key is not set. Its value should come from attributes in the incoming packet and is typically set by scripts. For example, CLID can be used to set the value of Session-Key.

Use the function UseCLIDAsSessionKey as defined in the script rexscript.c to specify that the Calling-Station-Id attribute that should be used as the session key to correlate requests for the same session. This is a typical case for 3G mobile user session correlation. You can provide your own script to define other attributes as the session key.

In the absence of the Session-Key variable, the key to the session will be created based on the string concatenated by the value of the NAS-Identifier and the NAS-Port.

There is a new option with-key available in aregcmd for query-sessions and release-sessions to access sessions by Session-Key.

Resource Managers

Resource Managers allow you to allocate dynamic resources to user sessions. The following lists the different types of Resource Managers.

- **IP-Dynamic**—manages a pool of IP addresses that allows you to dynamically allocate IP addresses from a pool of addresses
- **IP-Per-NAS-Port**—allows you to associate ports to specific IP addresses, and thus ensure each NAS port always gets the same IP address
- **IPX-Dynamic**—manages a pool of IPX network addresses
- **Subnet-Dynamic**—manages a pool of subnet addresses
- **Group-Session-Limit**—manages concurrent sessions for a group of users; that is, it keeps track of how many sessions are active and denies new sessions after the configured limit has been reached
- **User-Session-Limit**—manages per-user concurrent sessions; that is, it keeps track of how many sessions each user has and denies the user a new session after the configured limit has been reached
- **Home-Agent**—manages a pool of on-demand IP addresses
- **USR-VPN**—manages Virtual Private Networks (VPNs) that use USR NAS Clients.
- **Home-Agent-IPv6**—manages a pool of on-demand IPv6 addresses
- **Remote-IP-Dynamic**—manages a pool of IP addresses that allows you to dynamically allocate IP addresses from a pool of addresses. It internally works with a remote ODBC database.
- **Remote-User-Session-Limit**—manages per-user concurrent sessions; that is, it keeps track of how many sessions each user has and denies the user a new session after the configured limit has been reached. It internally works with a remote ODBC database.

- **Remote-Group-Session-Limit**—manages concurrent sessions for a group of users; that is, it keeps track of how many sessions are active and denies new sessions after the configured limit has been reached. It internally works with a remote ODBC database.

- **3GPP**—allows you to define the attribute for 3GPP authorization.

Each Resource Manager is responsible for examining the request and deciding whether to allocate a resource for the user, do nothing, or cause Cisco Prime Access Registrar to reject the request.

Table 4-24 lists the Resource Manager properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Resource Managers list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the Resource Manager.</td>
</tr>
</tbody>
</table>

### Types of Resource Managers

A number of different types of Resource Managers exist that allow you to manage IP addresses dynamically or statically, limit sessions on a per group or per user basis, or manage a Virtual Private Network. See Appendix A, “Cisco Prime Access Registrar Tcl, REX, and Java Dictionaries,” for information on how to override these individual Resource Managers.

**Note**

Resource Manager supports the following remote type session managers: remote-ipdynamic, remote-session-cache, home-agent, remote-user-session-limit, home-agent-ipv6 and remote-group-session-limit.

This section contains the following topics:

- Group-Session-Limit
- Home-Agent
- Home-Agent-IPv6
- IP-Dynamic
- IP-Per-NAS-Port
- IPX-Dynamic
- Session-Cache
- Subnet-Dynamic
- User-Session-Limit
Group-Session-Limit

Group-Session-Limit allows you to manage concurrent sessions for a group of users; that is, it keeps track of how many sessions are active and denies new sessions after the configured limit has been reached.

When you use this Resource Manager, you must set the GroupSessionLimit property to the maximum number of concurrent sessions for all users.

Home-Agent

Home-Agent is a resource manager that supports dynamic HA assignment. You configure the home-agent resource manager with a list of IP addresses. The Prime Access Registrar server assigns those addresses to clients whose request dictionary has the right attributes to indicate that an assignment should be done. This is similar to the ip-dynamic resource manager.

Unlike the ip-dynamic resource manager, HAs are not exclusively allocated to an individual session but are shared among a set of sessions.

Detailed configuration information for the Home-Agent resource manager is found in Chapter 20, “Wireless Support.” When you use this Resource Manager, you must set the Home-Agent-IPAddresses property to a single IP address or a range of IP addresses.

Home-Agent-IPv6

Home-Agent-Ipv6 is a new resource manager used to configure IPv6 address.

IP-Dynamic

IP-Dynamic allows you to manage a pool of IP addresses from which you dynamically allocate IP addresses.

When you use the IP-Dynamic Resource Manager, provide values for the properties listed in Table 4-25.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetMask</td>
<td>Required; must be set to a valid net mask.</td>
</tr>
<tr>
<td>IPAddresses</td>
<td>Required; must be a list of IP address ranges.</td>
</tr>
</tbody>
</table>
### IP-Per-NAS-Port

**IP-Per-NAS-Port** allows you to associate specific IP addresses with specific NAS ports and thus ensures each NAS port always gets the same IP address.

When you use this Resource Manager, provide values for the properties listed in Table 4-26.

**Note** You must have the same number of IP addresses and ports.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetMask</td>
<td>Required; if used, must be set to a valid net mask.</td>
</tr>
<tr>
<td>NAS</td>
<td>Required; must be the name of a known Client. This value must be the same as the NAS-Identifier attribute in the Access-Request packet.</td>
</tr>
<tr>
<td>IPAddresses</td>
<td>Required; must be a list of IP address ranges.</td>
</tr>
<tr>
<td>NASPorts</td>
<td>Required list of NAS ports.</td>
</tr>
</tbody>
</table>

### IPX-Dynamic

**An IPX-Dynamic** Resource Manager allows you to dynamically manage a pool of IPX networks. When you use the IPX-Dynamic Resource Manager, you must set the Networks property to a valid set of numbers which correspond to your networks.

**Note** You cannot use IPX network number 0x0. If you attempt to configure a Resource Manager with an IPX network number of 0x0, validation will fail.

### Session-Cache

The **session-cache** Resource Manager supports the Identity Cache feature. You use session-cache Resource Managers to define the RADIUS attributes to store in cache. Set the QueryKey property to the XML attribute you want to key on such as XML-Address-format-IPv4 and list all attributes to be cached in the AttributesToBeCached subdirectory. Use the QueryMappings subdirectory to map XML attributes to RADIUS attributes.
Note
Session record size is limited by the operating system (OS) paging size (4 KB in Linux). If a request triggers creation of a session that exceeds the OS paging size, the request will be dropped and the session will not be created.

If the disk partition where Prime Access Registrar stores session backing store data (usually the disk partition where Prime Access Registrar is installed, such as /opt/CSCOar) is full, the subsequent packets that try to create sessions will be dropped and no sessions will be created due to lack of disk space.

Subnet-Dynamic

The subnet-dynamic Resource Manager supports the On Demand Address Pool feature. You use subnet-dynamic resource managers to provide pools of subnet addresses. Following is an example of the configuration of a subnet dynamic resource manager:

```
/Radius/ResourceManagers/newResourceMgr
Name = newResourceMgr
Description =
Type = subnet-dynamic
Subnet-Mask = 255.255.255.0
SubnetAddresses/
   10.1.0.0-10.1.10.0
   11.1.0.0-11.1.10.0
```

When you use the subnet-dynamic Resource Manager, provide values for the properties listed in Table 4-28.

### Table 4-27 Session-Cache Resource Manager Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QueryKey</td>
<td>Required; set the QueryKey to the a RADIUS attribute you want to key on, such as Framed-IP-Address. A change made in Prime Access Registrar requires that this attribute not be an XML attribute, even if this session-cache resource manager is being used for an XML query. <strong>Note</strong> Any existing session-cache resource managers using an XML attribute for the Query Key must be changed to a RADIUS attribute that this XML attribute is mapped to under QueryMappings.</td>
</tr>
<tr>
<td>PendingRemovalDelay</td>
<td>Required; length of time information remains in the cache after the session ends (defaults to 10 seconds)</td>
</tr>
<tr>
<td>AttributesToBeCached</td>
<td>Required; use this subdirectory to provide a list of RADIUS attributes you want to store in cache</td>
</tr>
<tr>
<td>QueryMappings</td>
<td>Required; list of attribute pairs, mapping the XML attributes on the left-hand side to the RADIUS attribute on the right-hand side.</td>
</tr>
</tbody>
</table>

**Note**
Table 4-28  Subnet-Dynamic Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Required</td>
</tr>
<tr>
<td>Subnet mask</td>
<td>Required; must be set to the size of the managed subnets</td>
</tr>
<tr>
<td>SubnetAddresses</td>
<td>Required; must be a valid range of IP addresses</td>
</tr>
</tbody>
</table>

User-Session-Limit

User-Session-Limit allows you to manage per-user concurrent sessions; that is, it keeps track of how many sessions each user has and denies the user a new session after the configured limit has been reached.

When you use the user-session-limit Resource Manager, set the user-session-limit property to the maximum number of concurrent sessions for a particular user.

USR-VPN

USR-VPN allows you to set up a Virtual Private Network (VPN) using a US Robotics NAS.

When you use this Resource Manager, provide values for the properties listed in Table 4-29.

Table 4-29  USR-VPN Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Identifier</td>
<td>Required; must be set to the VPN ID the USR NAS will use to identify a VPN.</td>
</tr>
<tr>
<td>Neighbor</td>
<td>Optional; if set, should be the IP address of the next hop router for the VPN.</td>
</tr>
<tr>
<td>FramedRouting</td>
<td>Optional; if set, should be <strong>RIP V2 Off</strong> or <strong>RIP V2 On</strong> if the USR NAS is to run RIP Version 2 for the user.</td>
</tr>
<tr>
<td>Gateway Name</td>
<td>Required; must be unique in the Gateways list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the gateway.</td>
</tr>
<tr>
<td>IPAddress</td>
<td>Required; IP address of the gateway.</td>
</tr>
<tr>
<td>SharedSecret</td>
<td>Required; must match the shared secret of the gateway.</td>
</tr>
<tr>
<td>TunnelRefresh</td>
<td>Optional; if specified it is the number of seconds the tunnel stays active before a secure “keepalive” is exchanged between the tunnel peers in order to maintain the tunnel open.</td>
</tr>
<tr>
<td>LocationID</td>
<td>Optional; if specified it is a string indicating the physical location of the gateway.</td>
</tr>
</tbody>
</table>
Dynamic-DNS

Prime Access Registrar supports the Dynamic DNS protocol providing the ability to update DNS servers.

When you use this Resource Manager, provide values for the properties listed in Table 4-30.

Table 4-30  DYNAMIC-DNS Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max DNS TTLS</td>
<td>Set the maximum TTL of the DNS record.</td>
</tr>
<tr>
<td>DNS Host bytes</td>
<td>Set the number of bytes to be used to construct the reverse zone entry.</td>
</tr>
<tr>
<td>Forward Zone Name</td>
<td>Set the name of the forward zone. For a given Resource Manager you must decide which forward zone you will be updating for sessions the resource manager will manage.</td>
</tr>
<tr>
<td>Reverse Zone Name</td>
<td>Set the name of the reverse zone.</td>
</tr>
<tr>
<td>Forward Zone Server</td>
<td>Set the Server IP of the forward zone.</td>
</tr>
<tr>
<td>Reverse Zone Server</td>
<td>Set the Server IP of the reverse zone.</td>
</tr>
<tr>
<td>Forward Zone TSIG KeyS</td>
<td>Server-wide security key to process all forward zone dynamic DNS updates. This is used if a ForwardZoneTSIGKey was not specified on the Resource Manager.</td>
</tr>
<tr>
<td>Reverse Zone TSIG Keys</td>
<td>Server-wide security key to process all reverse zone dynamic DNS updates. This is used if a ReverseZoneTSIGKey was not specified on the Resource Manager.</td>
</tr>
</tbody>
</table>

Remote-IP-Dynamic

The configuration is same as IP-Dynamic but internally it works with a remote ODBC database.

Remote-User-Session-Limit

The configuration is same as User-Session-Limit but internally it works with a remote ODBC database.

Remote-Group-Session-Limit

The configuration is same as Group-Session-Limit but internally it works with a remote ODBC database.

Remote-Session-Cache

The configuration is same as Session-Cache but it should be used with session manager of type remote.

3GPP

Prime Access Registrar provides a resource manager for 3GPP authorization. When you use this Resource Manager, provide values for the properties listed in Table 4-31.
Profiles

You use Profiles to group RADIUS attributes that belong together, such as attributes that are appropriate for a particular class of PPP or Telnet user. You can reference profiles by name from either the UserGroup or the User properties. Thus, if the specifications of a particular profile change, you can make the change in a single place and have it propagated throughout your user community.

Although you can use UserGroups or Profiles in a similar manner, choosing whether to use one rather than the other depends on your site. When you require some choice in determining how to authorize or authenticate a user session, then creating specific profiles, and creating a group that uses a script to choose among them is more flexible.

In such a situation, you might create a default group, and then write a script that selects the appropriate profile based on the specific request. The benefit to this technique is each user can have a single entry, and use the appropriate profile depending on the way they log in.

Table 4-32 lists the Profile properties.

Attributes

Attributes are specific RADIUS components of requests and responses defined in the Request and Response Attribute dictionaries. Use the arecmd command set to assign values to attributes.

For a complete list of the attributes, see Appendix C, “RADIUS Attributes.” When setting a value for a STRING-type attribute such as Connect-Info (which starts with an integer), you must use the hexadecimal representation of the integer. For example, to set the attribute Connect-Info to a value of 7:7, use a set command like the following:

    set Connect-Info 37:3A:37

<table>
<thead>
<tr>
<th>Table 4-31 3GPP Resource Manager Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fields</strong></td>
</tr>
<tr>
<td>EnableRegistrationFlow</td>
</tr>
<tr>
<td>EnableSessionTermination</td>
</tr>
<tr>
<td>ReuseExistingSession</td>
</tr>
<tr>
<td>HSSProxyService</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 4-32 Profile Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Property</strong></td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Attributes</td>
</tr>
</tbody>
</table>
Translations

Translations add new attributes to a packet or change an existing attribute from one value to another. The Translations subdirectory lists all definitions of Translations the RADIUS server can apply to certain packets.

Under the /Radius/Translations directory, any translation to insert, substitute, or translate attributes can be added. The following is a sample configuration under the /Radius/Translations directory:

```
  cd /Radius/Translations
  Add T1
  cd T1
  Set DeleAttrs Session-Timeout, Called-Station-Id
  cd Attributes
  Set Calling-Station-Id 18009998888
```

DeleAttrs is the set of attributes to be deleted from the packet. Each attribute is comma separated and no spaces are allowed between attributes. All attribute value pairs under the attributes subdirectory are the attributes and values that are going to be added or translated to the packet.

Under the /Radius/Translations/T1/Attributes directory, inserted or translated attribute value pairs can be set. These attribute value pairs are either added to the packet or replaced with the new value.

If a translation applies to an Access-Request packet, by referencing the definition of that translation, the Prime Access Registrar server modifies the Request dictionary and inserts, filters and substitutes the attributes accordingly. You can set many translations for one packet and the Prime Access Registrar server applies these translations sequentially.

---

**Note**

Later translations can overwrite previous translations.
Table 4-33 lists the Translation properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Translations list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the Translation</td>
</tr>
<tr>
<td>DeleteAttrs</td>
<td>Optional; lists attributes to be filtered out</td>
</tr>
</tbody>
</table>

TranslationGroups

You can add translation groups for different user groups under TranslationGroups. All Translations under the Translations subdirectory are applied to those packets that fall into the groups. The groups are integrated with the Prime Access Registrar Rule engine.

The Prime Access Registrar Administrator can use any RADIUS attribute to determine the Translation Group. The incoming and outgoing translation group can be different translation groups. For example, you can set one translation group for incoming translations and one for outgoing translations.

Under the /Radius/TranslationGroups directory, translations can be grouped and applied to certain sets of packets, which are referred to in a rule. The following is a sample configuration under the /Radius/TranslationGroups directory:

```
  cd /Radius/TranslationGroups
  Add CiscoIncoming
  cd CiscoIncoming
  cd Translations
  Set 1 T1
```

The translation group is referenced through the Prime Access Registrar Policy Engine in the /Radius/Rules/<RuleName>/Attributes directory. Incoming-Translation-Groups are set to a translation group (for example CiscoIncoming) and Outgoing-Translation-Groups to another translation group (for example CiscoOutgoing). Table 4-34 lists the Translation Group properties.

Table 4-34 TranslationGroups Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Translations list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the Translation Group</td>
</tr>
<tr>
<td>Translations</td>
<td>Lists of translation</td>
</tr>
</tbody>
</table>

Remote Servers

You can use the RemoteServers object to specify the properties of the remote servers to which Services proxy requests. RemoteServers are referenced by name from the RemoteServers list in either the radius, ldap or tacacs-udp Services.

Note
You must not configure a remote server with an IP address, which is same as that of the client. This is applicable for all types of remote servers.
Table 4-35 lists the common RemoteServers properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the RemoteServers list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the remote server.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Required; specifies the remote server protocol which can be radius, ldap, or tacacs-udp.</td>
</tr>
<tr>
<td>IPAddress</td>
<td>Required; this property specifies where to send the proxy request. It is the address of the remote server. You must set it to a valid IP address. The IP address format is enhanced to support IPv6 apart from IPv4 only for the RADIUS type remote server.</td>
</tr>
<tr>
<td>Port</td>
<td>Required; the port to which Cisco Prime Access Registrar sends proxy requests. You must specify a number greater than zero. If there is no default port number, you must supply the correct port number for your remote server. If you set a port to zero, Prime Access Registrar sets the port to the default value for the type of remote server being configured. For example, the following remote servers have these default port values: dynamic-dns—53 radius—1812 ldap—389 accounting—1813</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Required; the amount of time (in milliseconds) to wait before retrying a remote server that was offline. You must specify a number greater than zero. The default is 300,000 (5 minutes).</td>
</tr>
</tbody>
</table>

## Types of Protocols

The Remote Server protocol you specify determines what additional information you must provide. The following are the protocols available in Prime Access Registrar with their required and optional fields.

Prime Access Registrar provides the following RemoteServer protocol types:

- Dynamic DNS
- LDAP
- Map-Gateway
- Sigtran
- ODBC
- ODBC-Accounting
- OCI
- OCI-Accounting
- Prepaid-CRB
- Prepaid-IS835C
Dynamic DNS

The `dynamic-dns` RemoteServer is used with the Dynamic DNS feature. The following is the default configuration of a `dynamic-dns` RemoteServer.

```plaintext
[ //localhost/Radius/RemoteServers/ddns ]
Name = ddns
Description =
Protocol = dynamic-dns
IPAddress =
Port = 53
MaxTries = 3
InitialTimeout = 2000
MaxDNSRenamingRetries = 3
TrimHostName = TRUE
ForwardZoneTSIGKey =
ReverseZoneTSIGKey =
```

Table 4-36 lists and defines the `dynamic-dns` RemoteServer properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPAddress</td>
<td>The IPAddress address of the DNS server</td>
</tr>
<tr>
<td>Port</td>
<td>Port 53 is the port that most DNS servers will use as a default</td>
</tr>
<tr>
<td>MaxTries</td>
<td>Number of times the server tries to send dynamic updates to a DNS server</td>
</tr>
<tr>
<td>InitialTimeout</td>
<td>Time, in milliseconds, that the server waits for a response before retrying a dynamic DNS request</td>
</tr>
<tr>
<td>MaxRenamingRetries</td>
<td>Number of times that the <code>dynamic-dns</code> resource managers can try to add a host in DNS even if it detects that the host's name is already present. This controls the number of times <code>Prime Access Registrar</code> tries to modify a host's name to resolve a conflict on each failed update.</td>
</tr>
<tr>
<td>TrimHostName</td>
<td>Controls whether <code>Prime Access Registrar</code> trims the hostname string to the first period character (used to update dynamic DNS update records and to return the hostname option to clients). If this attribute is enabled, the hostname is truncated before the period. If disabled, the server retains the period characters in the hostname.</td>
</tr>
<tr>
<td>ForwardZoneTSIGKey</td>
<td>Server-wide security key to process all forward zone dynamic DNS updates. This is used if a ForwardZoneTSIGKey was not specified on the Resource Manager.</td>
</tr>
<tr>
<td>ForwardZoneTSIGKey</td>
<td>Server-wide security key to process all forward zone dynamic DNS updates. This is used if a ForwardZoneTSIGKey was not specified on the Resource Manager.</td>
</tr>
<tr>
<td>ReverseZoneTSIGKey</td>
<td>Server-wide security key to process all reverse zone dynamic DNS updates. This is used if a ReverseZoneTSIGKey was not specified on the Resource Manager.</td>
</tr>
</tbody>
</table>
LDAP

ldap specifies an LDAP server. When you specify the ldap protocol, provide the information listed in Table 4-37.

For any LDAP remote service, the server might perform the environment mappings at any time. This means that if the service is set to either authentication and authorization, authentication-only, or authorization-only, environment mappings will take place. RADIUS mappings will take place only if the service is set to perform authorization. Checkitem mappings will take place only if the service is set to perform authentication. Previously environment mappings only occurred when the service was set for both authentication and authorization. RADIUS mappings, environment mappings, and checkitem mappings will not take place, if bind-based authentication is enabled.

Table 4-37 ldap RemoteServer Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
<td>Required; defaults to port 389.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Required; the default is 15. The timeout property indicates how many seconds the RADIUS server will wait for a response from the LDAP server.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> Use InitialTimeout from above as a template, except this is timeout is specified in seconds.</td>
</tr>
<tr>
<td>HostName</td>
<td>Required; the LDAP server’s hostname or IP address.</td>
</tr>
<tr>
<td>BindName</td>
<td>Optional; the distinguished name (dn) to use when establishing a connection between the LDAP and RADIUS servers.</td>
</tr>
<tr>
<td>BindPassword</td>
<td>Optional; the password associated with the BindName.</td>
</tr>
<tr>
<td>SearchPath (Overridden by Search-Path environment variable)</td>
<td>Required; the path that indicates where in the LDAP database to start the search for user information.</td>
</tr>
<tr>
<td>Filter</td>
<td>Required; this specifies the search filter Cisco Prime Access Registrar uses when querying the LDAP server for user information. When you configure this property, use the notation “%s” to indicate where the user ID should be inserted. For example, a typical value for this property is “(uid=%s),” which means that when querying for information about user joe, use the filter uid=joe.</td>
</tr>
<tr>
<td>UserPasswordAttribute</td>
<td>Required; this specifies which LDAP field the RADIUS server should check for the user’s password.</td>
</tr>
<tr>
<td>LimitOutstandingRequests</td>
<td>Required; the default is FALSE. Cisco Prime Access Registrar uses this property in conjunction with the MaxOutstandingRequests property to tune the RADIUS server’s use of the LDAP server. When you set this property to TRUE, the number of outstanding requests for this RemoteServer is limited to the value you specified in MaxOutstandingRequests. When the number of requests exceeds this number, Cisco Prime Access Registrar queues the remaining requests, and sends them as soon as the number of outstanding requests drops to this number.</td>
</tr>
</tbody>
</table>
Remote Servers

MaxOutstandingRequests | Required when you have set the LimitOutstandingRequests to TRUE. The number you specify, which must be greater than zero, determines the maximum number of outstanding requests allowed for this remote server.

MaxReferrals | Required; must be a number equal to or greater than zero. This property indicates how many referrals are allowed when looking up user information. When you set this property to zero, no referrals are allowed.

Cisco Prime Access Registrar manages referrals by allowing the RADIUS server’s administrator to indicate an LDAP “referral attribute,” which might or might not appear in the user information returned from an LDAP query. When this information is returned from a query, Cisco Prime Access Registrar assumes it is a referral and initiates another query based on the referral. Referrals can also contain referrals.

Note | This is an LDAP v2 referral property.

ReferralAttribute | Required when you have specified a MaxReferrals value. This property specifies which LDAP attribute, returned from an LDAP search, to check for referral information.

Note | This is an LDAP v2 referral property.

ReferralFilter | Required when you have specified a MaxReferral value. This is the filter Cisco Prime Access Registrar uses when processing referrals. When checking referrals, the information Cisco Prime Access Registrar finds in the referral itself is considered to be the search path and this property provides the filter. The syntax is the same as that of the Filter property.

Note | This is an LDAP v2 referral property.

PasswordEncryptionStyle | The default is None. You can also specify crypt, dynamic, SHA-1, and SSHA-1.

EscapeSpecialCharInUserName | FALSE by default

DNSLookupAndLDAPRebindInterval | Specifies the timeout period after which the Prime Access Registrar server will attempt to resolve the LDAP hostname to IP address (DNS resolution); 0 by default

DataSourceConnections | Specifies the number of concurrent connections to the LDAP server. The default value is 8.

SearchScope | Specifies how deep to search within a search path; default is SubTree which indicates a search of the base object and the entire subtree of which the base object distinguished name is the highest object.

Base indicates a search of the base object only.

OneLevel indicates a search of objects immediately subordinate to the base object, but does not include the base object.

---

### Table 4-37  ldap RemoteServer Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxOutstandingRequests</td>
<td>Required when you have set the LimitOutstandingRequests to TRUE. The number you specify, which must be greater than zero, determines the maximum number of outstanding requests allowed for this remote server.</td>
</tr>
<tr>
<td>MaxReferrals</td>
<td>Required; must be a number equal to or greater than zero. This property indicates how many referrals are allowed when looking up user information. When you set this property to zero, no referrals are allowed. Cisco Prime Access Registrar manages referrals by allowing the RADIUS server’s administrator to indicate an LDAP “referral attribute,” which might or might not appear in the user information returned from an LDAP query. When this information is returned from a query, Cisco Prime Access Registrar assumes it is a referral and initiates another query based on the referral. Referrals can also contain referrals. Note: This is an LDAP v2 referral property.</td>
</tr>
<tr>
<td>ReferralAttribute</td>
<td>Required when you have specified a MaxReferrals value. This property specifies which LDAP attribute, returned from an LDAP search, to check for referral information. Note: This is an LDAP v2 referral property.</td>
</tr>
<tr>
<td>ReferralFilter</td>
<td>Required when you have specified a MaxReferral value. This is the filter Cisco Prime Access Registrar uses when processing referrals. When checking referrals, the information Cisco Prime Access Registrar finds in the referral itself is considered to be the search path and this property provides the filter. The syntax is the same as that of the Filter property. Note: This is an LDAP v2 referral property.</td>
</tr>
<tr>
<td>PasswordEncryptionStyle</td>
<td>The default is None. You can also specify crypt, dynamic, SHA-1, and SSHA-1.</td>
</tr>
<tr>
<td>EscapeSpecialCharInUserName</td>
<td>FALSE by default</td>
</tr>
<tr>
<td>DNSLookupAndLDAPRebindInterval</td>
<td>Specifies the timeout period after which the Prime Access Registrar server will attempt to resolve the LDAP hostname to IP address (DNS resolution); 0 by default</td>
</tr>
<tr>
<td>DataSourceConnections</td>
<td>Specifies the number of concurrent connections to the LDAP server. The default value is 8.</td>
</tr>
<tr>
<td>SearchScope</td>
<td>Specifies how deep to search within a search path; default is SubTree which indicates a search of the base object and the entire subtree of which the base object distinguished name is the highest object. Base indicates a search of the base object only. OneLevel indicates a search of objects immediately subordinate to the base object, but does not include the base object.</td>
</tr>
</tbody>
</table>
Map-Gateway

The following is the default configuration of a map gateway RemoteServer.

```plaintext
[ //localhost/Radius/RemoteServers/map-gateway ]
Name = map-gateway
Description =
Protocol = map-gateway
IPAddress =
Port = 0

LDAPToRadiusMappings Optional; a list of name/value pairs in which the name is the name of the ldap attribute to retrieve from the user record, and the value is the name of the RADIUS attribute to set to the value of the ldap attribute retrieved.

For example, when the LDAPToRadiusMappings has the entry: FramedIPAddress = Framed-IP-Address, the RemoteServer retrieves the FramedIPAddress attribute from the ldap user entry for the specified user, uses the value returned, and sets the Response variable Framed-IP-Address to that value.

LDAPToEnvironmentMappings Optional; a list of name/value pairs in which the name is the name of the ldap attribute to retrieve from the user record, and the value is the name of the Environment variable to set to the value of the ldap attribute retrieved.

For example, when the LDAPToEnvironmentMappings has the entry: group = User-Group, the RemoteServer retrieves the group attribute from the ldap user entry for the specified user, uses the value returned, and sets the Environment variable User-Group to that value.

LDAPToCheckItemMappings Optional; a list of LDAP attribute/value pairs which must be present in the RADIUS access request and must match, both name and value, for the check to pass.

For example, when the LDAPToCheckItemMappings has the entry: group = User-Group, the Access Request must contain the attribute group, and it must be set to User-Group.

UseSSL A boolean field indicating whether you want Cisco Prime Access Registrar to use SSL (Secure Socket Layer) when communicating with this RemoteServer. When you set it to TRUE, be sure to specify the CertificateDBPath field in the Advanced section, and be sure the port you specified for this RemoteServer is the SSL port used by the LDAP server.

UseBinaryPasswordComparison A boolean field that enables binary password comparison for authentication. This property when set to TRUE, enables binary password comparison. By default, this property is set to FALSE.

UseBindBasedAuthentication A boolean field that enables bind-based authentication with LDAP server. This property when set to TRUE, enables bind-based authentication. By default, this property is set to FALSE. When set to FALSE, it uses existing legacy authentication method.
ReactivateTimerInterval = 300000
SharedSecret =
MaxTries = 3
InitialTimeout = 2000

Sigtran

The following is the default configuration of a Sigtran RemoteServer.

```
[ //localhost/Radius/RemoteServers/rs ]
    Name = rs
    Description =
    Protocol = sigtran
    HostName =
    LocalSubSystemNumber =
    CgPAGlobalTitleAddress =
    SetOPCInCgPA =
    GlobalTitleTranslationScript~ =
    SUAConfigurationFilename =
    ReactivateTimerInterval =
    Timeout = 5000
    LimitOutstandingRequests = FALSE
    MaxOutstandingRequests = 0
```

The RPM packages such as lksctp-tools-1.0.10-1, lksctp-tools-doc-1.0.10-1 and lksctp-tools-devel-1.0.10-1 should be installed in Linux 5.3 before configuring sigtran remote server which eventually adds the scpt libs (libscpt.so.1.0.10).

The following files can be downloaded from http://lksctp.sourceforge.net/
- lksctp-tools-1.0.10-1.i386.rpm
- lksctp-tools-devel-1.0.10-1.i386.rpm
- lksctp-tools-doc-1.0.10-1.i386.rpm

Prime Access Registrar supports only:
- one object of Remoteserver with protocol type "sigtran"
- MAP version 3 (3GPP TS 29.002 V6.4.0 (2003-12) ) and ITU Q.773 TCAP

Only one Quintets is fetched from HLR. The ITU TCAP continue message is not supported.

Table 4-38 lists and defines the Sigtran RemoteServer properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HostName</td>
<td>Required; represents the IP address of remote Signalling Gateway specified in the SUAConfiguration file.</td>
</tr>
<tr>
<td>LocalSubSystemNumber</td>
<td>Required; the default value for this property is 0. This represents the subsystem number used by SUA user.</td>
</tr>
<tr>
<td>CgPAGlobalTitleAddress</td>
<td>Required; represents the Global Title Address of CallingPartyAddress.</td>
</tr>
<tr>
<td>SetOPCInCgPA</td>
<td>Required; if it is set to TRUE, OPC will be used in CallingPartyAddress.</td>
</tr>
</tbody>
</table>
Table 4-38  Sigtran RemoteServer Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global TitleTranslationScript</td>
<td>This is used to specify the name of script which is responsible for</td>
</tr>
<tr>
<td></td>
<td>translating IMSI to GTA.</td>
</tr>
<tr>
<td>SUAConfigurationFilename</td>
<td>Required; used to specify the name of configuration file for SUA</td>
</tr>
<tr>
<td></td>
<td>stack initialization.</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Required; represents the reactivate time interval to re-connect after</td>
</tr>
<tr>
<td></td>
<td>failure.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Required; represents the how long the remote server should wait</td>
</tr>
<tr>
<td></td>
<td>before marking the request as timedout.</td>
</tr>
<tr>
<td>LimitOutstandingRequests</td>
<td>Limits the outstanding request to HLR when it is set to TRUE.</td>
</tr>
<tr>
<td>MaxOutstandingRequests</td>
<td>This represents the maximum outstanding request to HLR.</td>
</tr>
</tbody>
</table>

**Note**
You should restart the Prime Access Registrar server, if you change any SIGTRAN related configuration.

**ODBC**

odbc specifies an ODBC server. Cisco Prime Access Registrar provides a RemoteServer object (and a service) to support Open Database Connectivity (ODBC), an open specification that provides application developers a vendor-independent API with which to access data sources. Table 4-39 lists the odbc server attributes.

For any ODBC remote service, the server might perform the environment mappings at any time. This means that if the service is set to either authentication and authorization, authentication-only, or authorization-only, environment mappings will take place. RADIUS mappings will take place only if the service is set to perform authorization. Checkitem mappings will take place only if the service is set to perform authentication. Previously environment mappings only occurred when the service was set for both authentication and authorization.

Table 4-39  odbc Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeout</td>
<td>Required; the default is 15. The timeout property indicates how many seconds the RADIUS server will wait for a response from the LDAP server.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> Use InitialTimeout from above as a template, except this is timeout is specified in seconds.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Must be set to odbc.</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Required; default is 300,000 milliseconds. Length of time to wait before attempting to reconnect if a thread is not connected to a data source.</td>
</tr>
</tbody>
</table>
### Table 4-39  odbc Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Connections</td>
<td>Required; default is 8. This represents the total number of connections Prime Access Registrar can open with the ODBC server; total number of threads Prime Access Registrar can create for the ODBC server.</td>
</tr>
<tr>
<td>ODBCDataSource</td>
<td>Required; defines all items required for the <code>odbc.ini</code> file. The Prime Access Registrar server automatically creates the <code>odbc.ini</code> file based on these settings.</td>
</tr>
<tr>
<td>SQLDefinition</td>
<td>SQLDefinition properties define the SQL you want to execute.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Type</strong>—<code>query</code> (Prime Access Registrar supports only type <code>query</code>).</td>
</tr>
<tr>
<td></td>
<td>- <strong>SQL</strong>—SQL query used to acquire the password</td>
</tr>
<tr>
<td></td>
<td>- <strong>UserPasswordAttribute</strong>—Defines the database column name for the user’s password.</td>
</tr>
<tr>
<td></td>
<td>- <strong>MarkerList</strong>—Defines all markers for the query. MarkerList uses the format <code>UserName/SQL_DATA_TYPE</code>.</td>
</tr>
<tr>
<td>ODBCToRadiusMappings</td>
<td>Optional; a list of name/value pairs in which the name is the name of the <code>odbc</code> attribute to retrieve from the user record, and the value is the name of the RADIUS attribute to set to the value of the <code>odbc</code> attribute retrieved.</td>
</tr>
<tr>
<td></td>
<td>For example, when the <strong>ODBCToRadiusMappings</strong> has the entry: <code>FramedIPAddress = Framed-IP-Address</code>, the RemoteServer retrieves the <code>FramedIPAddress</code> attribute from the <code>odbc</code> user entry for the specified user, uses the value returned, and sets the Response variable <code>Framed-IP-Address</code> to that value.</td>
</tr>
<tr>
<td>ODBCToEnvironmentMappings</td>
<td>Optional; a list of name/value pairs in which the name is the name of the <code>odbc</code> attribute to retrieve from the user record, and the value is the name of the Environment variable to set to the value of the <code>odbc</code> attribute retrieved.</td>
</tr>
<tr>
<td></td>
<td>For example, when the <strong>ODBCToEnvironmentMappings</strong> has the entry: <code>group = User-Group</code>, the RemoteServer retrieves the <code>group</code> attribute from the <code>odbc</code> user entry for the specified user, uses the value returned, and sets the Environment variable <code>User-Group</code> to that value.</td>
</tr>
<tr>
<td>ODBCToCheckItemMappings</td>
<td>Optional; a list of ODBC <code>attribute/value</code> pairs which must be present in the RADIUS access request and must match, both name and value, for the check to pass.</td>
</tr>
<tr>
<td></td>
<td>For example, when the <strong>ODBCToCheckItemMappings</strong> has the entry: <code>group = User-Group</code>, the Access Request must contain the attribute <code>group</code>, and it must be set to <code>User-Group</code>.</td>
</tr>
</tbody>
</table>
**ODBC-Accounting**

If you use the Oracle Accounting feature, you must configure an ODBC-Accounting RemoteServer object. *Table 4-40* lists and defines the ODBC-Accounting RemoteServer properties.

**Table 4-40  ODBC-Accounting RemoteServer Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the remote server; this property is mandatory, and there is no default</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of server</td>
</tr>
<tr>
<td>Protocol</td>
<td>Must be set to odbc-accounting</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Mandatory time interval (in milliseconds) to activate an inactive server; defaults to 300000 ms.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Mandatory time interval (in seconds) to wait for SQL operation to complete; defaults to 15 seconds</td>
</tr>
<tr>
<td>DataSourceConnections</td>
<td>Mandatory number of connections to be established; defaults to 8</td>
</tr>
<tr>
<td>ODBCDataSource</td>
<td>Name of the ODBCDataSource to use and must refer to one entry in the list of ODBC datasources configured under /Radius/Advanced/ODBCDataSources. Mandatory; no default</td>
</tr>
<tr>
<td>KeepAliveTimerInterval</td>
<td>Mandatory time interval (in milliseconds) to send a keepalive to keep the idle connection active; defaults to zero (0) meaning the option is disabled</td>
</tr>
<tr>
<td>BufferAccountingPackets</td>
<td>Mandatory, TRUE or FALSE, determines whether to buffer the accounting packets to local file, defaults to TRUE which means that packet buffering is enabled.</td>
</tr>
</tbody>
</table>

*Note*  When set to TRUE, a constant flow of incoming accounting packets can fill the buffer backing store files in /cisco-ar/data/odbc beyond the size configured in MaximumBufferFileSize. Configure BackingStoreDiscThreshold in /Radius/Advanced when using ODBC accounting. See Advanced, page 4-58 for information about how to configure BackingStoreDiscThreshold.

| MaximumBufferFileSize     | Mandatory if BufferAccountingPackets is set to TRUE, determines the maximum buffer file size, defaults to 10 Megabyte |
| NumberOfRetriesForBufferedPacket | Mandatory if BufferAccountingPackets is set to TRUE. A number greater than zero determines the number of attempts to be made to insert the buffered packet into Oracle. Defaults to 3. |

**OCI**

OCI service can be used to authenticate and authorize an access request by querying user information through OCI and to insert accounting records into a data store through OCI. For more information on OCI server properties, see ODBC/OCI, page 3-121.
OCI-Accounting

If you use the Oracle Accounting feature, you must configure an OCI-Accounting RemoteServer object. For more information, see ODBC/OCI-Accounting, page 3-123.

Prepaid-CRB

The following is the default configuration of a prepaid-crb RemoteServer. The Filename property is the name of the required shared library provided by the billing vendor. See CRB Prepaid Billing, page 16-7 for more information on Prepaid-CRB.

```plaintext
[ //localhost/Radius/RemoteServers/prepaid-crb ]
Name = prepaid-crb
Description =
Protocol = prepaid-crb
IPAddress =
Port = 0
Filename =
Connections = 8
```

Prepaid-IS835C

The following is the default configuration of a prepaid-is835c RemoteServer. The Filename property is the name of the required shared library provided by the billing vendor. See IS835C Prepaid Billing, page 16-2 for more information on Prepaid-IS835C.

```plaintext
[ //localhost/Radius/RemoteServers/prepaid-is835c ]
Name = prepaid-is835c
Description =
Protocol = prepaid-is835c
IPAddress =
Port = 0
Filename =
Connections = 8
```

RADIUS

`radius` specifies a RADIUS server. When you specify the `radius` protocol, supply the information in Table 4-41.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SharedSecret</td>
<td>Required; the secret shared between the remote server and the RADIUS server.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional; when set, must be the name of a known incoming script. Cisco Prime Access Registrar runs the IncomingScript after it receives the response.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional; when set, must be the name of a known outgoing script. Cisco Prime Access Registrar runs the OutgoingScript just before it sends the proxy request to the remote server.</td>
</tr>
<tr>
<td>Vendor</td>
<td>Optional; when set, must be the name of a known Vendor.</td>
</tr>
</tbody>
</table>
Chapter 4  Cisco Prime Access Registrar Server Objects

Rules

Prime Access Registrar supports SIGTRAN-M3UA to fetch the authentication vectors from HLR, which is required for EAP-AKA/EAP-SIM authentication. For more information on SIGTRAN-M3UA protocol, see Chapter 23, “SIGTRAN-M3UA”.

Rules

A Rule is a function that selects services based on all input information used by the function.

Advanced

Advanced objects let you configure system-level properties and the Attribute dictionary. Under normal system operation, you should not need to change the system-level properties.

Note

The notation required means Cisco Prime Access Registrar needs a value for this property. For most of these properties, you can use system defaults.

### Table 4-41  RADIUS Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxTries</td>
<td>Required; the number of times to send a proxy request to a remote server before deciding the server is offline. You must specify a number greater than zero. The default is 3.</td>
</tr>
<tr>
<td>InitialTimeout</td>
<td>Required: represents the number of milliseconds used as a timeout for the first attempt to send a specific packet to a remote server. For each successive retry on the same packet, the previous timeout value used is doubled. You must specify a number greater than zero. The default value is 2000 (or 2 seconds).</td>
</tr>
<tr>
<td>ACKAccounting</td>
<td>When ACKAccounting is TRUE, the Prime Access Registrar server waits for the Accounting-Response from the remote RADIUS server before sending the corresponding Accounting-Response to the client. When ACKAccounting is FALSE, the Prime Access Registrar server does not wait for the Accounting-Response and immediately returns an Accounting-Response to the client.</td>
</tr>
<tr>
<td>SendandForget</td>
<td>This field is available if the AcknowledgeAccounting option is disabled. After forwarding a proxy packet to the remote server and an initial response to the client, Prime Access Registrar maintains a buffer of the original request and a copy of the proxy request until it receives a response from the remote server or packet timeout is triggered. If SendandForget is enabled, Prime Access Registrar deletes the original and proxy requests from the buffer after sending the response to the client. This helps in reducing buffer pool exhaustion in case of a low-responding remote server.</td>
</tr>
</tbody>
</table>

SIGTRAN-M3UA

Prime Access Registrar supports SIGTRAN-M3UA to fetch the authentication vectors from HLR, which is required for EAP-AKA/EAP-SIM authentication. For more information on SIGTRAN-M3UA protocol, see Chapter 23, “SIGTRAN-M3UA”.

Note

The notation required means Cisco Prime Access Registrar needs a value for this property. For most of these properties, you can use system defaults.
Table 4-42 lists the **Advanced** properties.

**Table 4-42   Advanced Object Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LogServerActivity</td>
<td>Required; the default is FALSE, which means Cisco Prime Access Registrar logs all responses except Access-Accepts and Access-Challenges. Accepting the default reduces the load on the server by reducing that amount of information it must log. Note, the client is probably sending accounting requests to an accounting server, so the Access-Accept requests are being indirectly logged. When you set it to TRUE, Cisco Prime Access Registrar logs all responses to the server log file.</td>
</tr>
<tr>
<td>TLSv1Enabled</td>
<td>Applicable only for Diameter; Set to TRUE to use TLS version 1.0 and above for Diameter connection. Set to FALSE to use TLS version greater than 1.0 for Diameter connection.</td>
</tr>
<tr>
<td>MaximumNumberOfRadiusPackets</td>
<td>Required; the default is 8192. This is a critical property you should set high enough to allow for the maximum number of simultaneous requests. When more requests come in than there are packets allocated, Cisco Prime Access Registrar will drop those additional requests.</td>
</tr>
<tr>
<td>UDPPacketSize</td>
<td>Required; the default is 4096. RFC 2138 specifies the maximum packet length can be 4096 bytes. Do not change this value.</td>
</tr>
<tr>
<td>NumberOfRemoteUDPServerSockets</td>
<td>Required; the default value for this property is 4. The NumberOfRemoteUDPServerSockets property allows you to configure the number of source ports used while proxying requests to a remote radius server. If the NumberOfRemoteUDPServerSockets property is set to a value n, all remote servers share and use n sockets. The NumberOfRemoteUDPServerSockets value comprises a number, as in n, where n should be less than or equal to the current process file descriptor limit divided by 4.</td>
</tr>
</tbody>
</table>

**Note** By default, the RADIUS process supports up to 1024 file descriptors. To increase the file descriptors, stop the arserver; in the arserver script, specify the required value to "NUMBER_OF_FILE_DESCRIPTOR" and restart the server. The value for "NUMBER_OF_FILE_DESCRIPTOR" should be in the range between 1024 to 65535.
### Table 4-42 Advanced Object Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NumberOfRadiusIdentifiersPerSocket</td>
<td>This represents the number of RADIUS Identifiers that Prime Access Registrar can use per source port, while proxying requests to remote servers. To use a different source port for every request that is proxied, you need to set the value of this property to one.</td>
</tr>
<tr>
<td>MemoryLimitForRadiusProcess</td>
<td>This property is used to avoid crashing of the radius process. The default value is 3500 Megabytes. This property is under <code>/radius/advanced</code>. When the radius process uses memory more than the configured limit, further sessions are not created and Prime Access Registrar rejects further incoming requests.</td>
</tr>
<tr>
<td>MemorySizeCheckInterval</td>
<td>This property is used to avoid crashing of the radius process. This is used in conjunction with MemoryLimitForRadiusProcess. The default value is 5 minutes. MemorySizeCheckInterval is a hidden parameter in mcd database. To modify the default value, you need to export the mcd database. Typically, a separate thread is created to monitor the radius process memory usage for every 5 minutes.</td>
</tr>
<tr>
<td>PerPacketHeapSize</td>
<td>Required; the default is 6500. This property sets the size of the initial heap for each packet. The heap is the dynamic memory a request can use during its lifetime. By preallocating the heap size at the beginning of request processing, we can minimize the cost of memory allocations. If PerPacketHeapSize is too low, Prime Access Registrar will ask the system for memory more often. If PerPacketHeapSize is too high, Prime Access Registrar will allocate too much memory for the request causing the system to use more memory than required.</td>
</tr>
<tr>
<td>RequireNASsBehindProxyBeInClientList</td>
<td>Required; the default is FALSE. If you accept the default, Cisco Prime Access Registrar only uses the source IP address to identify the immediate client that sent the request. Leaving it FALSE is useful when this RADIUS Server should only know about the proxy server and should treat requests as if they came from the proxy server. This might be the case with some environments that buy bulk dial service from a third party and thus do not need to, or are unable to, list all of the NASs behind the third party’s proxy server. When you set it to TRUE, you must list all of the NASs behind the Proxy in the Clients list. For more information about this property, see Using the RequireNASsBehindProxyBeInClientList Property, page 4-75.</td>
</tr>
</tbody>
</table>
Table 4-42 Advanced Object Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAAFileServiceSyncInterval</td>
<td>Required; specified in milliseconds, the default is 75. This property governs how often the file AAA service processes accounting requests and writes the accounting records to the file. You can lower the number to reduce the delay in acknowledging the Account-Request at the expense of more frequent flushing of the accounting file to disk. You can raise the number to reduce the cost of flushing to disk, at the expense of increasing the delays in acknowledging the Accounting-Requests. The default value was determined to provide a reasonable compromise between the two alternatives.</td>
</tr>
<tr>
<td>SessionBackingStoreSyncInterval</td>
<td>Required; specified in milliseconds, the default is 100. If you change this value it must be a number greater than zero. This property governs how often the Session Manager backing store writes updated session information to disk. You can lower the number to reduce the delay in acknowledging requests at the expense of more frequent flushing of the file containing the session data to disk. You can raise the number to reduce the cost of flushing to disk at the expense of increasing delays in acknowledging requests. The default value was determined to provide a reasonable compromise between the two alternatives.</td>
</tr>
<tr>
<td>BackingStoreDiscThreshold</td>
<td>Required; the default is 10 gigabytes. The value of BackingStoreDiscThreshold is made up of a number of units which can be K, kilobyte, or kilobytes, M, megabyte, or megabytes, or G, gigabyte, or gigabytes. BackingStoreDiscThreshold is used with session management and ODBC accounting and ensures that any data log files generated will not cross the BackingStoreDiscThreshold.</td>
</tr>
<tr>
<td>SessionBackingStorePruneInterval</td>
<td>Required; specifies the sleep time interval of the session backing store pruning thread. The recommended and default value is 6 hours, but you can modify this based on the traffic patterns you experience. With SessionBackingStorePruneInterval set to 6 hours, pruning will occur 6 hours after you restart or reload the Prime Access Registrar server and recur every 6 hours. You can set a very low value for this property to make pruning continuous, but there might not be enough data accumulated for the pruning to occur and pruning might be less effective compared to the default setting.</td>
</tr>
</tbody>
</table>
### PacketBackingStorePruneInterval

Required; specifies the sleep time interval of the packet backing store pruning thread. The recommended value is 6 hours, but you can modify this based on the traffic patterns you experience.

When PacketBackingStorePruneInterval is set to 6 hours, pruning will occur 6 hours after you restart or reload the Prime Access Registrar server and recur every 6 hours.

You can set a very low value for this property to make pruning continuous, but there might not be enough data accumulated for the pruning to occur and pruning might be less effective compared to the default setting.

### RemoteLDAPServerThreadTimerInterval

Required; specified in milliseconds, the default is 10. This property governs how often the ldap RemoteServer thread checks to see if any results have arrived from the remote LDAP server. You can modify it to improve the throughput of the server when it proxies requests to a remote LDAP server.

### RemoteSigtranServerThreadTimerInterval

Required; specified in milliseconds, the default is 10. This property governs how often the sigtran RemoteServer thread checks to see if any results have arrived from the remote HLR/AuC server. You can modify it to improve the throughput of the server when it proxies requests to a remote sigtran server.

### InitialBackgroundTimerSleepTime

Required; the default is 5. This property specifies the amount of time the time queue should initially sleep before beginning processing. This property is only used for initial synchronization and should not be changed.

### MinimumSocketBufferSize

Required; the default is 65536 (64 K). This property governs how deep the system’s buffer size is for queueing UDP datagrams until Cisco Prime Access Registrar can read and process them. The default is probably sufficient for most sites. You can, however, raise or lower it as necessary.

### CertificateDBPath

Required if you are using an LDAP RemoteServer and you want Prime Access Registrar to use SSL when communicating with that LDAP RemoteServer. This property specifies the path to the directory containing the client certificates to be used when establishing an SSL connection to an LDAP RemoteServer. This directory must contain the cert7.db and cert5.db certificates and the key3.db and key.db files database used by Netscape Navigator 3.x (and above) or the ServerCert.db certificate database used by Netscape 2.x servers.
### Table 4-42  Advanced Object Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LogFileSize</td>
<td>Required; the default is 1 Megabyte. This property specifies the maximum size of the RADIUS server log file. The value for the LogFileSize field is a string composed of two parts; a number, and a units indicator (&lt;n&gt; &lt;units&gt;) in which the unit is one of: K, kilobyte, kilobytes, M, megabyte, megabytes, G, gigabyte, or gigabytes. The LogFileSize property does not apply to the config_mcd_1_log or agent_server_1_log files. See Modifying File Sizes for Agent Server and MCD Server Logs, page 29-3 to configure these files.</td>
</tr>
<tr>
<td></td>
<td>Note  This does not apply to the trace log.</td>
</tr>
<tr>
<td>LogFileCount</td>
<td>Required; the default is 2. This property specifies the number of log files to be kept on the system. A new log file is created when the log file size reaches LogFileSize. The LogFileCount property does not apply to the config_mcd_1_log or agent_server_1_log files. See Modifying File Sizes for Agent Server and MCD Server Logs, page 29-3 to configure these files.</td>
</tr>
<tr>
<td>TraceFileSize</td>
<td>Required; the default is 1 GB. This property specifies the size of the trace files to be kept on the system. A new trace file is created when the trace file size reaches TraceFileSize. The value for the TraceFileSize field is a string composed of two parts; a number, and a units indicator (&lt;n&gt; &lt;units&gt;) in which the unit is one of: K, kilobyte, kilobytes, M, megabyte, megabytes, G, gigabyte, or gigabytes.</td>
</tr>
<tr>
<td>TraceFileCount</td>
<td>Required; this value can be set from 1-100, and the default is 2. This property specifies the number of trace files to maintain. A value of 1 indicates that no file rolling occurs.</td>
</tr>
<tr>
<td>UseAdvancedDuplicateDetection</td>
<td>Required; the default is FALSE. Set this property to TRUE when you want Cisco Prime Access Registrar to use a more robust duplicate request filtering algorithm. For more information on this property, see Advance Duplicate Detection Feature, page 4-75.</td>
</tr>
<tr>
<td>AdvancedDuplicateDetectionMemoryInterval</td>
<td>Required when the Advanced Duplicate Detection feature is enabled. This property specifies how long (in milliseconds) Cisco Prime Access Registrar should remember a request. You must specify a number greater than zero. The default is 10,000.</td>
</tr>
</tbody>
</table>
DetectOutOfOrderAccountingPackets

Optional; used to detect accounting packets that arrive out of sequential order. The default is FALSE. This property is useful when using accounting and session management in a RADIUS proxy service.

When the DetectOutOfOrderAccountingPacket property is enabled (set to TRUE), a new Class attribute is included in all outgoing Accept packets. The value for this Class attribute will contain the session magic number. The client will echo this value in the accounting packets, and this will be used for comparison.

The session magic number is a unique number created for all sessions when the session is created or reused and the DetectOutOfOrderAccountingPacket property is set to TRUE. The DetectOutOfOrderAccountingPacket property is used to detect out-of-order Accounting-Stop packets in roaming scenarios by comparing the session magic number value in the session with the session magic number value contained in the Accounting packet.

The value of 0xffffffff is considered by the Prime Access Registrar server to be a wildcard magic number. If any accounting stop packets contain the value of 0xffffffff, it will pass the session magic validation even if the session’s magic number is something else.

The format of the class attribute is as follows:

\(<4\text{-byte Magic Prefix}>\text{<4-byte server IP address><4-byte Magic value}>\)

DefaultReturnedSubnetSizeIfNoMatch

Optional; used with the ODAP feature and reflects the returned size of the subnet if no matched subnet is found. There are three options to select if an exactly matched subnet does not exist: Bigger, Smaller, and Exact. The default is Bigger.

ClasspathForJavaExtensions

A string which is the classpath to be used to locate Java classes and jar files containing the classes required for loading the Java extensions, either Java extension points or services.

Note: The classpath will always contain the directory $INSTALLDIR/scripts/radius/java and all of the jar files in that directory.

JavaVMOptions

A string that can contain options to be passed to the JRE upon startup. JavaVMOptions should be used only when requested by Cisco TAC.

MaximumODBCResultSize

Specifies maximum size in bytes for an ODBC mapping. This parameter affects both ODBC result sizes and the trace log buffer for tracing script calls that access any of the dictionaries. (Default value is 256.)
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARIsCaseInsensitive</td>
<td>When set to FALSE, requires that you provide exact pathnames with regard to upper and lower case for all objects, subobjects, and properties. The default setting, TRUE, allows you to enter paths such as /rad/serv instead of /Rad/Serv.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> Prime Access Registrar always authenticates the RADIUS attribute User-Name with regard to upper and lower case, regardless of the setting of this flag.</td>
</tr>
<tr>
<td>RemoteRadiusServerInterface</td>
<td>When set, specifies the local interface to bind to when creating the RemoteRadiusServer socket. If not set, the Prime Access Registrar binds to IPADDR_ANY.</td>
</tr>
<tr>
<td>ODBCEnvironmentMultiValueDelimiter</td>
<td>Optional; allows you to specify a character that separates multivalued attributes in the marker list when using Oracle (or ODBC) accounting.</td>
</tr>
<tr>
<td>PacketBackingStoreSyncInterval</td>
<td>The minimum value is 1 and the maximum is a 32-bit unsigned integer. The default is 75.</td>
</tr>
<tr>
<td>ListenForDynamicAuthorizationRequests</td>
<td>Must be set to TRUE when using the Change of Authorization (CoA) feature or Packet of Disconnect (POD) feature. Default is FALSE.</td>
</tr>
<tr>
<td>MaximumNumberOfXMLPackets</td>
<td>Required when using identity caching. Indicates the maximum number of XML packets to be sent or received. The minimum value is 1 and the maximum is a 32-bit unsigned integer. The default is 1024.</td>
</tr>
<tr>
<td>XMLUDPPacketSize</td>
<td>Required when using identity caching. Indicates the maximum size of XML packets to be sent or received. The minimum value is 1 and the maximum is a 32-bit unsigned integer. The default is 4096.</td>
</tr>
<tr>
<td>RollingEncryptionKeyChangePeriod</td>
<td>Used in conjunction with the session-cache ResourceManager, this property specifies the length of time a given EncryptionKey will be used before a new one is created. When the session-cache ResourceManager caches User-Password attributes, Prime Access Registrar encrypts the User-Password so it is not stored in memory or persisted on disk in clear text. Prime Access Registrar uses up to 255 encryption keys, using a new one after each RollingEncryptionKeyChangePeriod expires. If RollingEncryptionKeyChangePeriod is set to 2 days, Prime Access Registrar will create and begin using a new EncryptionKey every two days. The oldest key will be retired, and Prime Access Registrar will re-encrypt any User-Passwords that used the old key with the new key. This way, if the RollingEncryptionKeyChangePeriod is set to 1 day, no key will be older than 255 days.</td>
</tr>
</tbody>
</table>
### Advanced Object Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| SessionPurgeInterval  | Optional; the SessionPurgeInterval property determines the time interval at which to check for timed-out sessions. If no value is set, the session timeout feature is disabled. The checks are performed in the background when system resources are available, so checks might not always occur at the exact time set.

The minimum recommended value for SessionPurgeInterval is 60 minutes. The SessionPurgeInterval value is comprised of a number and a units indicator, as in n units, where a unit is one of minutes, hours, days, or weeks. |
| EapBadMessagePolicy   | Set to one of two values: SilentDiscard (the default) or RejectFailure. If set to SilentDiscard, the Prime Access Registrar server silently discards and ignores bad EAP messages unless the protocol specification explicitly requires a failure message. When set to RejectFailure, the Prime Access Registrar server sends RADIUS Access-Rejects messages with embedded EAP-Failure in response to bad EAP messages as described in Internet RFC 3579. |
| StaleSessionTimeout   | Required; the default value is “1 hour.” Specifies the time interval to maintain a session when a client does not respond to Accounting-Stop notification. When the Prime Access Registrar server does not receive an Accounting-Response from a client after sending an Accounting-Stop packet, Prime Access Registrar maintains the session for the time interval configured in this property before releasing the session. This property is stored as a string composed of two parts: a number and a unit indicator (<n> <units>) similar to the MaxFileAge property where the unit is one of: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, or Weeks. |
| Ports                 | Optional; allows you to use ports other than the default, 1812 and 1813. You can use this option to configure Prime Access Registrar to use other ports. If you add additional ports, however, Prime Access Registrar will use the added ports and no longer use ports 1812 and 1813. These ports can still be used by adding them to the list of ports to use. For more information, see Ports, page 4-76. |
| Interfaces            | Optional; see Interfaces, page 4-76. |
| ReplyMessages         | Optional; see Reply Messages, page 4-76. |
| AttributeDictionary   | Optional; see Attribute Dictionary, page 4-78. |
| SNMP                  | Optional; see SNMP, page 4-79. |
### Table 4-42 Advanced Object Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFC Compliance</td>
<td>Optional; enables you to modify the Prime Access Registrar server to behave in a way that might deviate from RFC compliance in a special use case scenario. When AllowRejectAttrs is set to FALSE, Reply-Message attributes will not be passed in an Access Reject packet. When AllowRejectAttrs is set to TRUE, attributes will be allowed to pass in an Access Reject packet. When AllowEAPRejectAttrs is set to FALSE, Reply-Message attributes will not be passed in an Access Reject packet if the packet contains EAP-Message attribute. When AllowEAPRejectAttrs is set to TRUE, attributes will be allowed to pass in an Access Reject packet even if the packet contains EAP-Message attribute. <strong>Note</strong> Changing the state of either of these properties requires you to <strong>reload</strong> the Prime Access Registrar server.</td>
</tr>
<tr>
<td>DDNS</td>
<td>This subdirectory holds the SynthesizeReverseZone property and a list of Transaction Signatures (TSIG) keys.</td>
</tr>
<tr>
<td>SynthesizeReverseZone</td>
<td>This property exists under DDNS and controls whether Prime Access Registrar automatically generates the name of the reverse zone (<strong>in-addr.arpa</strong>) that is updated with PTR records. If this attribute is enabled and the resource manager does not have an explicit ReverseZoneName property configured, the server uses the IP address and DNSHostBytes property to generate the reverse zone name. The default value is TRUE.</td>
</tr>
<tr>
<td>ODBCDataSources</td>
<td>A list of ODBC data sets and their associated environments including operating system, DBMS, and network platform used to access the DBMS an application wants to access. Required when using Oracle (or ODBC) accounting.</td>
</tr>
<tr>
<td>AttributeGroups</td>
<td>Includes a Default subdirectory with an Attributes subdirectory that contains commonly-used attributes for Change of Authorization (CoA) and Packet of Disconnect (POD). You can add new attributes to the default group or create a new group as necessary.</td>
</tr>
<tr>
<td>KeyStores</td>
<td>Used to protect the security and integrity of the PACs it issues.</td>
</tr>
<tr>
<td></td>
<td>- NumberOfKeys—Number (from 1-1024) that specifies the maximum number of keys stored for EAP-FAST.</td>
</tr>
<tr>
<td></td>
<td>- RolloverPeriod—Specifies the amount of time between key updates.</td>
</tr>
<tr>
<td>DefaultRadiusSharedSecret</td>
<td>Enter the default shared secret for RADIUS server.</td>
</tr>
</tbody>
</table>
### Table 4-42 Advanced Object Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaximumIncomingRequestRate</td>
<td>Optional; the default value for this property is 0.</td>
</tr>
<tr>
<td></td>
<td>The MaximumIncomingRequestRate property is used to limit the incoming traffic in terms of “allowed requests per second”. Serves as a soft limit.</td>
</tr>
<tr>
<td></td>
<td>The MaximumIncomingRequestRate property comprises a number ( n ), where ( n ) can be any nonzero value.</td>
</tr>
<tr>
<td>HideSharedSecretAndPrivateKeys</td>
<td>Required; the default value is TRUE.</td>
</tr>
<tr>
<td></td>
<td>The HideSharedSecretAndPrivateKeys property hides:</td>
</tr>
<tr>
<td></td>
<td>• The secret that is shared between a RADIUS Client and a RADIUS Server or between two radius servers in a radius proxy scenario.</td>
</tr>
<tr>
<td></td>
<td>• The PrivateKeyPassword under the certificate-based EAP services.</td>
</tr>
<tr>
<td></td>
<td>When this property is set to TRUE, the following properties are displayed as &lt;encrypted&gt;:</td>
</tr>
<tr>
<td></td>
<td>• PrivateKeyPasswords in:</td>
</tr>
<tr>
<td></td>
<td>– peap-v0 service</td>
</tr>
<tr>
<td></td>
<td>– peap-v1 service</td>
</tr>
<tr>
<td></td>
<td>– eap-tls service</td>
</tr>
<tr>
<td></td>
<td>– eap-ttls service</td>
</tr>
<tr>
<td></td>
<td>– eap-fast service</td>
</tr>
<tr>
<td></td>
<td>• SharedSecret in:</td>
</tr>
<tr>
<td></td>
<td>– RemoteServers of type radius</td>
</tr>
<tr>
<td></td>
<td>– RemoteServers of type map-gateway</td>
</tr>
<tr>
<td></td>
<td>– Clients object</td>
</tr>
<tr>
<td></td>
<td>– Resource Manager of type usr-vpn under Gateway subobject</td>
</tr>
<tr>
<td></td>
<td>• PseudonymSecret in eap-sim service</td>
</tr>
<tr>
<td></td>
<td>• DynamicAuthSecret under DynamicAuthorizationServer subbject in Clients object</td>
</tr>
<tr>
<td></td>
<td>• RepSecret under Replication</td>
</tr>
<tr>
<td></td>
<td>• Secret in /radius/advanced/DDNS/TSIGKeys</td>
</tr>
</tbody>
</table>

When the value for this property is set to FALSE, all the above properties are displayed in clear text.
Table 4-42  Advanced Object Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| MaximumOutstandingRequests| Optional; the default value for this property is 0.  
The MaximumOutstandingRequests property is used to limit the incoming traffic in terms of “requests processed”. Serves as a hard limit.  
The MaximumOutstandingRequests property comprises a number n, where n can be any nonzero value. |
| Diameter                  | Required; See Diameter, page 4-80                                                                                                         |
### Table 4-42 Advanced Object Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPSSamplingPeriodInSecs</td>
<td>This represents the sampling period in seconds. The minimum sampling period is set to 5. The default is 30.</td>
</tr>
<tr>
<td>LogTPSActivity</td>
<td>When set to true this property enables to log in the TPS usage in a CSV file. The TPS is logged in the following format: (&lt;mm-dd-yyyy&gt;, <a href="">hh:mm:ss</a>, &lt;tps-value&gt;, &lt;sigtran-m3ua traffic value&gt;) For example, 04-24-2014,18:36:30,2998,1000 The default is False.</td>
</tr>
<tr>
<td>TPSLogFilenamePrefix</td>
<td>This represents the prefix of the CSV file which will be available in the logs directory of Prime Access Registrar. The following represents the CSV filename format: (&lt;user-prefix&gt;-&lt;mm-dd-yyyy&gt;.csv) tps-04-01-2013.csv</td>
</tr>
<tr>
<td>TPSLogFileCount</td>
<td>Configures the number of TPS Sampling log files to be maintained in the repository. The default value is 2.</td>
</tr>
<tr>
<td>LogSessionActivity</td>
<td>When set to TRUE, this property enables Prime Access Registrar to log the session count in the server.</td>
</tr>
<tr>
<td>EnableLengthFlag</td>
<td>Set to TRUE to enable the length flag.</td>
</tr>
<tr>
<td>SessionLogFileCount</td>
<td>Required only if you set LogSessionActivity to TRUE; the number of session log files to maintain in the repository. The default value is 2.</td>
</tr>
<tr>
<td>SessionLogFileNamePrefix</td>
<td>Required only if you set LogSessionActivity to TRUE; this represents the prefix of the session log file which will be available in the logs directory of Prime Access Registrar.</td>
</tr>
<tr>
<td>SessionSamplingPeriodInSecs</td>
<td>Required only if you set LogSessionActivity to TRUE; this represents the session sampling period in seconds. The minimum sampling period is set to 5. The default is 30.</td>
</tr>
<tr>
<td>FlushDiskInBackground</td>
<td>Set to TRUE to allow Prime Access Registrar to flush the accounting record to disk before it acknowledges the request packets.</td>
</tr>
</tbody>
</table>
Table 4-42  Advanced Object Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdditionalNativeOracleErrors</td>
<td>Optional; used to disconnect ODBC Remote Servers when configured native Oracle Error has occurred (which are not considered as connection errors). You must specify Native Errors as comma (,) separated integer values. For example, 04/14/2013 11:06:43.692: Log: ODBC client (DataSource 'CVOracleAcctDb', Connection 6): SQLExecute failed: SQLState:HY000 NativeError:12152 ErrorText:[Easysoft][Oracle]ORA-12152: TNS:unable to send break message 04/14/2013 10:44:59.388: Log: ODBC client (DataSource 'CVOracleAcctDb', Connection 3): SQLExecute failed: SQLState:HY000 NativeError:3114 ErrorText:[Easysoft][Oracle]ORA-03114: not connected to ORACLE For the above examples, the Native Errors need to be configured as follows: --&gt; set AdditionalNativeOracleErrors 12152,3114 When any one of the Native Errors 12152 or 3114 occurs, Prime Access Registrar disconnects the ODBC Remote Server.</td>
</tr>
<tr>
<td>SendOpCodeInISDResponse</td>
<td>Set to TRUE to send operator code in the ISD response.</td>
</tr>
<tr>
<td>EnableRoutingContextInM3UA</td>
<td>Set to TRUE to enable routing context in M3UA.</td>
</tr>
<tr>
<td>EnableSIGTRANStackLogs</td>
<td>When set to TRUE, this property enables to log the SIGTRAN stack logs in stack.log file.</td>
</tr>
<tr>
<td>SIGTRANStackLogFileSize</td>
<td>Required if you set EnableSIGTRANStackLogs to TRUE. This property specifies the maximum size (in megabyte) of the SIGTRAN stack log file.</td>
</tr>
<tr>
<td>SIGTRANLogFileCount</td>
<td>Required if you set EnableSIGTRANStackLogs to TRUE. This value can be set from 1–100, and the default is 10. This property specifies the number of SIGTRAN log files to maintain in the repository.</td>
</tr>
<tr>
<td>EnableStickySessionCount</td>
<td>Required; either True or False and the default value is True. When set to True, Prime Access Registrar displays the peer specific stats showing the number of sticky sessions associated with a peer for Diameter proxy service in name_radiusus_log file.</td>
</tr>
<tr>
<td>StickySessionCountInterval</td>
<td>Required; specified in milliseconds and the default is 60000. When the EnableStickySessionCount is set to True, this field specifies how often the Diameter proxy service will display the number of sticky sessions associated with a peer.</td>
</tr>
</tbody>
</table>
### Advanced Object Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>StickySessionSyncInterval</td>
<td>Required; specified in milliseconds and the default value is 500. Specifies how often the Diameter proxy service will write the sticky sessions to a file located in /cisco-ar/temp/__sticky_sessions_store location.</td>
</tr>
<tr>
<td>ReserveRADIUSPacketPool</td>
<td>Percentage of the RADIUS packet pool to reserve for the RADIUS remote server responses.</td>
</tr>
</tbody>
</table>

The following CLI shows an example configuration of Advanced properties:

```plaintext
LogServerActivity = FALSE
TLSv1Enabled = FALSE
MaximumNumberOfRadiusPackets = 8192
UDPPacketSize = 4096
NumberOfRemoteUDPServerSockets = 4
NumberOfRadiusIdentifiersPerSocket = 256
PerPacketHeapSize = 6500
RequireNASsBehindProxyBeInClientList = FALSE
AAAPfileServiceSyncInterval = 75
SessionBackingStoreSyncInterval = 100
BackingStoreDiscThreshold = "5 Gigabyte"
SessionBackingStorePruneInterval = "6 Hours"
PacketBackingStorePruneInterval = "6 Hours"
RemoteLDAPServerThreadTimerInterval = 10
RemoteSigtranServerThreadTimerInterval = 10
InitialBackgroundTimerSleepTime = 5
MinimumSocketBufferSize = 65536
CertificateDBPath =
LogFileSize = "1 Megabyte"
LogFileCount = 2
TraceFileSize = "1 Gigabyte"
TraceFileCount = 2
MemoryLimitForRadiusProcess = "3584 Megabyte"
UseAdvancedDuplicateDetection = FALSE
AdvancedDuplicateDetectionMemoryInterval = 10000
DetectOutOfOrderAccountingPackets = FALSE
DefaultReturnedSubnetSizeIfNoMatch = BIGGER
ClasspathForJavaExtensions =
JavaVMOptions =
MaximumODBCResultSize = 256
ARIsCaseInsensitive = TRUE
RemoteRadiusServerInterface =
ODBCEnvironmentMultiValueDelimiter =
PacketBackingStoreSyncInterval = 75
ListenForDynamicAuthorizationRequests = FALSE
MaximumNumberOfXMLPackets = 1024
XMLUDPPacketSize = 4096
RollingEncryptionKeyChangePeriod = "1 week"
SessionPurgeInterval =
EapBadMessagePolicy = SilentDiscard
StaleSessionTimeout = "1 Hour"
MaximumOutstandingRequests = 0
MaximumIncomingRequestRate = 0
HideSharedSecretAndPrivateKeys = TRUE
DefaultRadiusSharedSecret =
LogTSSActivity = TRUE
TPSLogFileCount = 15
TPSLogFilenamePrefix = tps
```
TPSSamplingPeriodInSecs = 15  
LogSessionActivity = TRUE  
EnableLengthFlag = FALSE  
SessionLogFileCount = 15  
SessionLogFilenamePrefix = sm  
SessionSamplingPeriodInSecs = 30  
FlushDiskInBackground = FALSE  
AdditionalNativeOracleErrors =  
SendOpCodeInISDResponse = FALSE  
EnableRoutingContextInM3UA = FALSE  
EnableStickySessionCount = TRUE  
EnableSIGTRANStackLogs = TRUE  
SIGTRANStackLogFileSize = "100 Megabyte"  
SIGTRANLogFileCount = 10  
StickySessionCountInterval = 500  
StickySessionSyncInterval = 500  
ReserveRADIUSPacketPool = 0  
Ports/  
Interfaces/  
ReplyMessages/  
Attribute Dictionary/  
SNMP/  
RemoteSessionServer/  
RFCCompliance/  
DDNS/  
ODBCDataSources/  
AttributeGroups/  
KeyStores/  
Diameter/  
DiameterDictionary/ 

This section contains the following topics:

- RemoteODBCSessionServer
- Using the RequireNASsBehindProxyBeInClientList Property
- Advance Duplicate Detection Feature
- Invalid EAP Packet Processing
- Ports
- Interfaces
- Reply Messages
- Attribute Dictionary
- SNMP
- Diameter

RemoteODBCSessionServer

The following is an example of the RemoteODBCSessionServer configuration:

--> cd /Radius/Advanced/RemoteODBCSessionServer/

[ //localhost/Radius/Advanced/RemoteODBCSessionServer ]  
ReactivateTimerInterval = 300000  
Timeout = 15  
DataSourceConnections = 8  
ODBCDataSource =
KeepAliveTimerInterval = 0  
BufferAccountingPackets = TRUE  
MaximumBufferFileSize = "10 Megabytes"  
CacheLimit = 250000  
UseCacheIndex = 0

Table 4-43 lists and defines the RemoteODBCSessionServer properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReactivateTimerInterval</td>
<td>Mandatory time interval (in milliseconds) to activate an inactive server; defaults to 300000 ms</td>
</tr>
<tr>
<td>Timeout</td>
<td>Mandatory time interval (in seconds) to wait for SQL operation to complete; defaults to 15 seconds</td>
</tr>
<tr>
<td>DataSourceConnections</td>
<td>Mandatory number of connections to be established; defaults to 8</td>
</tr>
<tr>
<td>ODBCDataSource</td>
<td>Name of the ODBCDataSource to use and must refer to one entry in the list of ODBC datasources configured under /Radius/Advanced/ODBCDataSources. Mandatory; no default</td>
</tr>
<tr>
<td>KeepAliveTimerInterval</td>
<td>Mandatory time interval (in milliseconds) to send a keepalive to keep the idle connection active; defaults to zero (0) meaning the option is disabled</td>
</tr>
</tbody>
</table>
| BufferAccountingPackets   | Mandatory, TRUE or FALSE, determines whether to buffer the accounting packets to local file, defaults to TRUE which means that packet buffering is enabled.  
**Note** When set to TRUE, a constant flow of incoming accounting packets can fill the buffer backing store files in /cisco-ar/data/odbc beyond the size configured in MaximumBufferFileSize. Configure BackingStoreDiscThreshold in /Radius/Advanced when using ODBC accounting. See Advanced, page 4-58 for information about how to configure BackingStoreDiscThreshold. |
| MaximumBufferFileSize     | Mandatory if BufferAccountingPackets is set to TRUE, determines the maximum buffer file size, defaults to 10 Megabyte) |
| CacheLimit                | Default is 250000; This represents the overall limit on cache of all 'remote' session managers. This value is interpreted as the maximum number of packets that can be present in cache. When the number of sessions hits this limit, sessions will be 'cached out'. This cache out operation will continue, until the cache is at least 20% free. |
| UseCacheIndex             | If set to 1, it enables a fast cache based lookup index for the items in the database. This optimizes the number of queries to the database hence will improve performance, but limits the number of sessions that can be scaled.  
If set to 0, it disables fast cache based lookup index. |

**Note** Remote session manager will work only with Oracle database.
Using the RequireNASsBehindProxyBeInClientList Property

You can use the property `RequireNASsBehindProxyBeInClientList` to require NASs that send requests indirectly through a proxy to be listed in the Clients list or to allow the proxy to represent them all.

- When you want to ensure the proxy is only sending requests from NASs known to this server, set the property to TRUE, and list all of the NASs using this proxy. This increases memory usage.
- When it is impossible to know all of the NASs using this proxy or when you do not care, set the property to FALSE. Cisco Prime Access Registrar will use the proxy’s IP address to identify the origin of the request.

Advance Duplicate Detection Feature

Prime Access Registrar automatically detects and handles duplicate requests it is currently working on. It also provides an optional, more complex mechanism to handle duplicate requests that can be received by the server after it has completed processing the original request. These duplicate requests can consume extra processing power, and, if received out of order (as RADIUS is a UDP-based protocol) might cause Session Management problems.

One solution is the Advanced Duplicate Detection feature which causes Prime Access Registrar to remember requests it has seen, as well as the response sent to that request, for a configurable amount of time.

To enable this feature, perform the following:

- Set the `UseAdvancedDuplicateDetection` property in the `/Radius/Advanced` section of the configuration to TRUE.
- Set the `AdvancedDuplicateDetectionMemoryInterval` in the `/Radius/Advanced` section to specify how long (in milliseconds) Prime Access Registrar should remember a request.

Note: Enabling this feature causes Cisco Prime Access Registrar to keep more of its preallocated packet buffers in use for a longer period of time. The number of preallocated buffers is controlled by the `MaximumNumberOfRadiusPackets` property in the `/Radius/Advanced` section of the configuration. This property might need to be increased (which will increase the amount of memory used by Cisco Prime Access Registrar) when the Advanced Duplicate Detection feature is enabled.

Invalid EAP Packet Processing

Prime Access Registrar has been enhanced to implement fatal error packet handling for Extensible Authentication Protocol (EAP) messages as described in section 2.2 of Internet RFC 3579 which states the following:

A RADIUS server determining that a fatal error has occurred must send an Access-Reject containing an EAP-Message attribute encapsulating EAP-Failure.

Because this enhancement is a deviation from various EAP specifications, you must explicitly enable this feature through a new configuration property in `/Radius/Advanced` named `EapBadMessagePolicy`. 
You can set the EapBadMessagePolicy property to one of two values: SilentDiscard (the default) or RejectFailure. When set to SilentDiscard, the Prime Access Registrar server silently discards and ignores bad EAP messages unless the protocol specification explicitly requires a failure message. When set to RejectFailure, the Prime Access Registrar server sends RADIUS Access-Rejects messages with embedded EAP-Failure in response to bad EAP messages as described in Internet RFC 3579.

The implementation of EAP authentication methods in Prime Access Registrar behaves as described in Internet RFC 2284 (EAP) and related EAP method specifications. These specify silent discard as the standard way to handle all EAP error conditions. Any EAP response message from the client that contains an error or is received in an invalid authenticator state is discarded and there is no error response.

In a configuration where EAP requests are proxied between RADIUS servers using RADIUS messages (EAP over RADIUS), the silent discard of an EAP message means that no RADIUS response message is sent back to the originating RADIUS server. Because of this, the RADIUS server originating the request eventually declares the destination RADIUS server dead and fails over to a backup server (if so configured).

### Ports

The Ports list specifies which ports to listen to for requests. When you specify a port, Cisco Prime Access Registrar makes no distinction between the port used to receive Access-Requests and the port used to receive Accounting-Requests. Either request can come in on either port.

Most NASs send Access-Requests to port 1812 and Accounting-Requests to 1813, however, Cisco Prime Access Registrar does not check.

When you do not specify any ports, Cisco Prime Access Registrar reads the `/etc/services` file for the ports to use for access and accounting requests. If none are defined, Prime Access Registrar uses the standard ports (1812 and 1813).

### Interfaces

The Interfaces list specifies the interfaces on which the RADIUS server receives and sends requests. You specify an interface by its IP address.

- When you list an IP address, Cisco Prime Access Registrar uses that interface to send and receive Access-Requests.
- When no interfaces are listed, the server performs an interface discover and uses all interfaces of the server, physical and logical (virtual).

**Note**

The IP address format is enhanced to support both IPv4 and IPv6.

### Reply Messages

The Reply Messages list allows you to choose the reply message based on the reason the request was rejected. Each of the following properties (except Default) corresponds to a reason why the packet was rejected. The Reply Message properties allows you to substitute your own text string for the defined errors. After you set the property (with the set command) and the reason occurs, Cisco Prime Access Registrar sends the NAS that message in the Access-Reject packet as a
**Reply-Message** attribute.

You might want to substitute your own messages to prevent users from getting too much information about why their requests failed. For example, you might not want users to know the password was invalid to prevent hackers from accessing your system. In such a case, you might specify the text string “unauthorized access” for the property **UserPasswordInvalid**.

Table 4-44 lists the **Reply Message** properties.

### Table 4-44  Reply Message Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends this value when the property corresponding to the reject reason is not set.</td>
</tr>
<tr>
<td>UnknownUser</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <strong>Reply-Message</strong> attribute whenever Cisco Prime Access Registrar cannot find the user specified by <strong>User-Name</strong>.</td>
</tr>
<tr>
<td>UserNotEnabled</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <strong>Reply-Message</strong> attribute whenever the user account is disabled.</td>
</tr>
<tr>
<td>UserPasswordInvalid</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <strong>Reply-Message</strong> attribute whenever the password in the Access-Request packet did not match the password in the database.</td>
</tr>
<tr>
<td>UnableToAcquireResource</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <strong>Reply-Message</strong> attribute whenever one of the Resource Managers was unable to allocate the resource for this request.</td>
</tr>
<tr>
<td>ServiceUnavailable</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <strong>Reply-Message</strong> attribute whenever a service the request needs (such as a RemoteServer) is unavailable.</td>
</tr>
<tr>
<td>InternalError</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <strong>Reply-Message</strong> attribute whenever an internal error caused the request to be rejected.</td>
</tr>
<tr>
<td>MalformedRequest</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <strong>Reply-Message</strong> attribute whenever a required attribute (such as <strong>User-Name</strong>) is missing from the request.</td>
</tr>
<tr>
<td>ConfigurationError</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <strong>Reply-Message</strong> attribute whenever the request is rejected due to a configuration error. For example, if a script sets an environment variable to the name of an object such as <strong>Authentication-Service</strong>, and that object does not exist in the configuration, the reason reported is ConfigurationError.</td>
</tr>
<tr>
<td>IncomingScriptFailed</td>
<td>Optional; when you set this property, Cisco Prime Access Registrar sends back this value in the <strong>Reply-Message</strong> attribute whenever one of the <strong>IncomingScripts</strong> fails to execute.</td>
</tr>
</tbody>
</table>
The Attribute dictionary allows you to specify the attributes to the RADIUS server. Cisco Prime Access Registrar comes with the standard RADIUS attributes (as defined by the RFC 2865) as well as the attributes required to support the major NASs. For more information about the standard attributes, see Appendix C, “RADIUS Attributes.”

All RADIUS requests and responses consist of one or more attributes, such as the user’s name, the user’s password, the type of service the NAS should provide to the user, or the IP address the user should use for the session.

In the request and response packets, an attribute is composed of a number (between 1-255) that specifies the type of attribute to use, a length that specifies the entire attribute length, and a value. How the value is interpreted depends on its type. When it is a username, the value is a string. When it is the NAS’s IP address, the value is an IP address, and so on.

Table 4-45 lists the Attribute dictionary properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Attribute dictionary list within the same context. Although it should be an attribute defined in the RFC, the name can be any attribute defined by your client. The NAS typically comes with a list of attributes it uses. Attributes are referenced in the Profile and by Scripts by this name. The accounting file service also uses this name when printing the attribute.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the attribute.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Required; must be a number between 1-255. It must be unique within the Attribute dictionary list.</td>
</tr>
<tr>
<td>Type</td>
<td>Required; must be set to one of the types listed in Table 4-46. The type governs how the value is interpreted and printed.</td>
</tr>
</tbody>
</table>
Types

Types are required and must be one of the following listed in Table 4-46.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNDEFINED</td>
<td>Treated as a string of binary bytes.</td>
</tr>
<tr>
<td>UINT32</td>
<td>Unsigned 32-bit integer.</td>
</tr>
<tr>
<td>STRING</td>
<td>Character string.</td>
</tr>
<tr>
<td>IPADDR</td>
<td>A valid IP address in dotted-decimal format.</td>
</tr>
<tr>
<td>CHAP_PASS</td>
<td>17-byte value representing the password.</td>
</tr>
<tr>
<td>ENUM</td>
<td>Enums allow you to specify the mapping between the value and the strings.</td>
</tr>
<tr>
<td></td>
<td>After you have established this mapping, Cisco Prime Access Registrar then</td>
</tr>
<tr>
<td></td>
<td>replaces the number with the appropriate string.</td>
</tr>
<tr>
<td></td>
<td>The min/max properties represent the lowest to highest values of the</td>
</tr>
<tr>
<td></td>
<td>enumeration.</td>
</tr>
<tr>
<td>VENDOR_SPECIFIC</td>
<td>Vendor Specific Attribute (VSAs) are a special class of attribute.</td>
</tr>
<tr>
<td></td>
<td>VSAs were created to extend the standard 256 attributes to include attributes</td>
</tr>
<tr>
<td></td>
<td>required by specific manufacturers.</td>
</tr>
<tr>
<td></td>
<td>VSAs add new capabilities for the value field in an attribute.</td>
</tr>
<tr>
<td></td>
<td>Rather than being a simple integer string, or IP address, the value of a</td>
</tr>
<tr>
<td></td>
<td>VSA can be one or more subattributes whose meaning depends on the vendor’s</td>
</tr>
<tr>
<td></td>
<td>definition. The Vendors list allows you to add, delete, or modify the</td>
</tr>
<tr>
<td></td>
<td>definitions of the vendors and the subattributes they specify.</td>
</tr>
</tbody>
</table>

Vendor Attributes

Table 4-47 lists the Vendor properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; must be unique in the Vendors attribute list.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the subattribute list.</td>
</tr>
<tr>
<td>VendorID</td>
<td>Required; must be a valid number and unique within the entire attribute</td>
</tr>
<tr>
<td></td>
<td>dictionary.</td>
</tr>
<tr>
<td>Type</td>
<td>Required; must be one of the following: UNDEFINED, UINT32, STRING, IPADDR,</td>
</tr>
<tr>
<td></td>
<td>CHAP_PASS, ENUM, or SUB_ATTRIBUTES.</td>
</tr>
</tbody>
</table>

SNMP

Table 4-48 lists the five properties of the SNMP directory.
Table 4-48  **SNMP Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enabled</td>
<td>Either TRUE or FALSE; default is FALSE</td>
</tr>
<tr>
<td>TracingEnabled</td>
<td>Either TRUE or FALSE; default is FALSE</td>
</tr>
<tr>
<td>InputQueueHighThreshold</td>
<td>An integer; default is 90</td>
</tr>
<tr>
<td>InputQueueLowThreshold</td>
<td>An integer; default is 60</td>
</tr>
<tr>
<td>MasterAgentEnabled</td>
<td>Either TRUE or FALSE; default is TRUE</td>
</tr>
</tbody>
</table>

If Enabled and MasterAgentEnabled are both TRUE, `arservagt` will start and stop the SNMP daemon (`snmpd`). If either of these properties is FALSE, if the Prime Access Registrar server is not using SNMP or if your site uses a different master agent, `arservagt` will not start your master agent.

### Diameter

This section explains how to configure Diameter general configuration and transport management.

Change the directory to `/Radius/Advanced/Diameter`.

```bash
//localhost/Radius/Advanced/Diameter
  General/
    TransportManagement/
```

The following configuration is used to configure Diameter general configuration like Product name and Version.

```bash
[ //localhost/Radius/Advanced/Diameter/General ]
  Product = Cisco Prime Access Registrar
  Version = 7.0
  AuthApplicationIdList = 1
  AcctApplicationIdList = 3
```

Table 4-49 describes the Diameter general properties.

Table 4-49  **Diameter General Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>Optional; name of the product.</td>
</tr>
<tr>
<td>Version</td>
<td>Optional; version number.</td>
</tr>
</tbody>
</table>
The following example shows the Diameter transport management configuration:

```
[ //localhost/Radius/Advanced/Diameter/TransportManagement ]
Identity = toby-ar1.cisco.com
Realm = cisco.com
TCPListenPort = 3868
SCTPListenPort = 3868
EnableIPV6 = FALSE
WatchdogTimeout = 500
ReconnectInterval = 500
MaxReconnections = 3
RequestRetransmissionInterval = 100
MaxRequestRetransmissionCount = 3
ReceiveBufferSize = 2048
AdvertisedHostNames = toby-ar1.cisco.com
```
Table 4-50 describes the Diameter transport management properties.

**Table 4-50  Diameter Transport Management Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identity</td>
<td>Required; identity of the system on which Diameter application is running. Must be set to a valid resolvable string.</td>
</tr>
<tr>
<td>Realm</td>
<td>Required; must be set to a valid Realm in the domain.</td>
</tr>
<tr>
<td>TCPListenPort</td>
<td>Required; port number on which Prime Access Registrar server listens for TCP peer connections.</td>
</tr>
<tr>
<td>SCTPListenPort</td>
<td>Required; port number on which Prime Access Registrar server listens for SCTP peer connections.</td>
</tr>
<tr>
<td>EnableIPv6</td>
<td>Required; if set to TRUE it enables IPV6 for the Diameter application.</td>
</tr>
<tr>
<td>WatchdogTimeout</td>
<td>Required; specifies the time interval between watch dog messages.</td>
</tr>
<tr>
<td>ReconnectInterval</td>
<td>Required; specifies the time interval between which Prime Access Registrar server attempts to connect to a disconnected peer. If set to 0, then no attempt will be made to connect to a disconnected peer.</td>
</tr>
<tr>
<td>MaxReconnections</td>
<td>Required; specifies the number of times Prime Access Registrar server tries to Make a reconnection attempt. If set to 0, then no attempt will be made to reconnect.</td>
</tr>
<tr>
<td>RequestRetransmissionInterval</td>
<td>Required; the time for which retransmission of pending requests will be done. If set to 0, then no attempt will be made to retransmit.</td>
</tr>
<tr>
<td>MaxRequestRetransmissionCount</td>
<td>Required, maximum number of times Prime Access Registrar server tries to retransmit a pending request. If set to 0, then no attempt will be made to retransmit.</td>
</tr>
<tr>
<td>ReceiveBufferSize</td>
<td>Required; initial size of buffer that is preallocated for message reception.</td>
</tr>
<tr>
<td>AdvertisedHostNames</td>
<td>Optional; specifies the local hostname address that will be advertised by the Prime Access Registrar server to other peers during CER/CEA exchange.</td>
</tr>
</tbody>
</table>

For example:

AdvertisedHostNames = toby-ar1.cisco.com
Configuring Diameter Session Management

The following example shows the Diameter session management configuration:

```
//localhost/Radius/Advanced/Diameter/SessionManagement
MaxSessions = 10000
AuthSessions/
AcctSessions/
AuthSessions/
EnableStatefulSessions = TRUE
AuthSessionTimeout = 5
LifeTimeTimeout = 360
GracePeriodTimeout = 30
AbortRetryTimeout = 20
AcctSessions/
AcctSessionTimeout = 30
InterimInterval = 5
RealTime = 0
```

Table 4-51 describes the Diameter Session Management properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxSessions</td>
<td>Required; specifies the maximum number of concurrent Diameter sessions Prime Access Registrar server will maintain. These sessions include both Auth and Acct sessions.</td>
</tr>
<tr>
<td>AuthSessions/EnableStatefulSessions</td>
<td>If set to TRUE, the server will enforce stateful sessions and the client will hint for stateful sessions. Default Value is TRUE. Set the property to FALSE to disable stateful sessions.</td>
</tr>
<tr>
<td>AuthSessionTimeout</td>
<td>Required; specifies the timeout in seconds before a session requires reauthentication.</td>
</tr>
<tr>
<td>LifeTimeTimeout</td>
<td>Required; specifies the timeout in seconds before a session is terminated regardless of whether the session has been reauthenticated.</td>
</tr>
<tr>
<td>GracePeriodTimeout</td>
<td>Required; specifies the grace period after the life timeout and before the full termination of the session.</td>
</tr>
<tr>
<td>AbortRetryTimeout</td>
<td>Required; specifies the timeout between the subsequent Abort Session Request (ASR) messages if the initial attempt fails.</td>
</tr>
<tr>
<td>AcctSessions/AcctSessionTimeout</td>
<td>Required; specifies the timeout in seconds before a session requires reauthentication.</td>
</tr>
<tr>
<td>InterimInterval</td>
<td>Required; specifies the interim interval dictated to the client if the entity is a server or hint to the server if the entity is a client.</td>
</tr>
<tr>
<td>RealTime</td>
<td>Required; RealTime value dictated to the client.</td>
</tr>
</tbody>
</table>
Configuring Diameter Application

Table 4-52 describes the Diameter Application properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; name of the application.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the application.</td>
</tr>
<tr>
<td>IsVendorSpecific</td>
<td>Required; the default is FALSE. If set to FALSE, the application is ordinary application. If set to TRUE, the application is a VendorSpecific Application.</td>
</tr>
<tr>
<td>IsAuthApplication</td>
<td>Required; if set to TRUE the application represents AuthApplication else it represents Accounting Application.</td>
</tr>
<tr>
<td>Application ID</td>
<td>Required; specifies the unique integer value for the application.</td>
</tr>
<tr>
<td>Application URI</td>
<td>Optional; specifies the URI of the Application.</td>
</tr>
</tbody>
</table>

The following are examples of Diameter application:
- NASREQ 1
- Mobile-IP 2
- Diameter Base Accounting 3

**Note** ApplicationId property must be set to 0 for Base Protocol.

<table>
<thead>
<tr>
<th>VendorSpecificApplicationID</th>
<th>Required; specifies the integer value for the vendor specific application.</th>
</tr>
</thead>
<tbody>
<tr>
<td>VendorID</td>
<td>Required; specifies the VendorID for the application.</td>
</tr>
<tr>
<td>ApplicationURI</td>
<td>Optional; specifies the URI of the Application.</td>
</tr>
<tr>
<td>Commands</td>
<td>Required; an indexed list from 1 to &lt;n&gt;. Each entry in the list is the name of the command. It specifies the list of commands associated with the application.</td>
</tr>
</tbody>
</table>

### Configuring the Diameter Application

To configure the Diameter application:

**Step 1** Move to //localhost/Radius/Advanced/Diameter/Applications directory:

**Step 2** Add the application you want to add (eg: NASREQ).

```
add NASREQ
```

Added NASREQ
cd NASREQ

[ //localhost/Radius/Advanced/Diameter/Applications/NASREQ ]
Name = NASREQ
Description =
IsAuthApplication = TRUE
IsVendorSpecific = FALSE
ApplicationID =
ApplicationURI =
Commands/

**Step 3** Set the ApplicationId and ApplicationURI.

```plaintext
set ApplicationId 1
Set ApplicationId 1
```

**Step 4** Add the list of commands for this application.

```plaintext
cd commands/
Set 1 AA
```

---

**Configuring Diameter Commands**

Table 4-53 describes the Diameter command properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; name of the command.</td>
</tr>
<tr>
<td>CommandCode</td>
<td>Required; specifies the integer code of the command.</td>
</tr>
<tr>
<td>EnableProxyBit</td>
<td>Required; default is TRUE. When enabled it represents the message is proxiable.</td>
</tr>
</tbody>
</table>
To configure the Diameter commands:

**Step 1** Change to /Radius/Advanced/Diameter/Commands.

**Step 2** Add AA command.

```
add AA

[ //localhost/Radius/Advanced/Diameter/Commands ]

cd AA/

cd AA/
```
Step 3   Set the properties for AA command.

    [ //localhost/Radius/Advanced/Diameter/Commands/AA ]
    Name = AA
    Description =
    CommandCode =
    EnableProxyBit = TRUE
    RequestMsgAVPs/
    AnswerMsgAVPs/

    set CommandCode 265

    Set CommandCode 265

    set EnableProxyBit TRUE

    Set EnableProxyBit TRUE

Step 4   Configure the RequestMsgAVP's for the command.

    cd RequestMsgAVPs/

    [ //localhost/Radius/Advanced/Diameter/Commands/AA/RequestMsgAVPs ]
    Fixed/
    Required/
    Optional/

    Add Fixed AVP's for the request message.

    Add Fixed AVP's

    cd Fixed/

    add Session-Id

    Added Session-Id

    cd Session-Id/

    [ //localhost/Radius/Advanced/Diameter/Commands/AA/RequestMsgAVPs/Fixed/Session-Id ]
    Name = Session-Id
    Description =
    Min = 0
    Max =

    Maximum and Minimum property specifies the multiplicity of the AVP Inside a request (or response). Similarly add the required and Optional AVP's.

Step 5   Configure AnswerMsgAVP's similar to step 3.

    cd AnswerMsgAVPs/

    [ //localhost/Radius/Advanced/Diameter/Commands/AA/AnswerMsgAVPs ]
    Fixed/
    Required/
    Optional/
The following shows an example of NASREQ application configuration:

```plaintext
[ //localhost/Radius/Advanced/Diameter/Applications/NASREQ ]
Name = NASREQ
Description =
IsAuthApplication = TRUE
IsVendorSpecific = FALSE
ApplicationID = 1
ApplicationURI =
Commands/
1. AA
```

The following shows an example of the AA command configuration:

```plaintext
[ //localhost/Radius/Advanced/Diameter/Commands ]
Entries 1 to 1 from 1 total entries
Current filter: <all>

[ //localhost/Radius/Advanced/Diameter/Commands/AA ]
Name = AA
Description =
CommandCode = 265
EnableProxyBit = TRUE
RequestMsgAVPs/
    Fixed/
        Entries 1 to 1 from 1 total entries
        Current filter: <all>

        Session-Id/
            Name = Session-Id
            Description =
            Min = 1
            Max = 1
        Required/
            Entries 1 to 7 from 7 total entries
            Current filter: <all>

        Auth-Application-Id/
            Name = Auth-Application-Id
            Description =
            Min = 1
            Max = 1
        Auth-Request-Type/
            Name = Auth-Request-Type
            Description =
            Min = 1
            Max = 1
        Destination-Realm/
            Name = Destination-Realm
            Description =
            Min = 1
            Max = 1
        Origin-Host/
            Name = Origin-Host
            Description =
            Min = 1
            Max = 1
        Origin-Realm/
            Name = Origin-Realm
            Description =
            Min = 1
            Max = 1
        User-Name/
            Name = User-Name
```
Description =
Min = 0
Max = 1
User-Password/
   Name = User-Password
   Description =
   Min = 0
   Max = 1
Optional/
   Entries 1 to 42 from 42 total entries
   Current filter: <all>
   ARAP-Password/
      Name = ARAP-Password
      Description =
      Min = 0
      Max = 1
   ARAP-Security/
      Name = ARAP-Security
      Description =
      Min = 0
      Max = 1
   ARAP-Security-Data/
      Name = ARAP-Security-Data
      Description =
      Min = 0
      Max = 100
Auth-Grace-Period/
   Name = Auth-Grace-Period
   Description =
   Min = 0
   Max = 1
Auth-Session-State/
   Name = Auth-Session-State
   Description =
   Min = 0
   Max = 1
Authorization-Lifetime/
   Name = Authorization-Lifetime
   Description =
   Min = 0
   Max = 1
AnswerMsgAVPs/
   Fixed/
      Entries 1 to 1 from 1 total entries
      Current filter: <all>
      Session-Id/
         Name = Session-Id
         Description =
         Min = 1
         Max = 1
Required/
   Entries 1 to 5 from 5 total entries
   Current filter: <all>
   Auth-Application-Id/
      Name = Auth-Application-Id
      Description =
      Min = 1
      Max = 1
   Auth-Request-Type/
      Name = Auth-Request-Type
      Description =
Min = 1
Max = 1
Origin-Host/
  Name = Origin-Host
  Description =
  Min = 1
  Max = 1
Origin-Realm/
  Name = Origin-Realm
  Description =
  Min = 1
  Max = 1
Result-Code/
  Name = Result-Code
  Description =
  Min = 1
  Max = 1
Optional/
  Entries 1 to 59 from 59 total entries
  Current filter: <all>

Acct-Interim-Interval/
  Name = Acct-Interim-Interval
  Description =
  Min = 0
  Max = 1
ARAP-Challenge-Response/
  Name = ARAP-Challenge-Response
  Description =
  Min = 0
  Max = 1
ARAP-Features/
  Name = ARAP-Features
  Description =
  Min = 0
  Max = 1
ARAP-Security/
  Name = ARAP-Security
  Description =
  Min = 0
  Max = 1
ARAP-Security-Data/
  Name = ARAP-Security-Data
  Description =
  Min = 0
  Max = 100
ARAP-Zone-Access/
  Name = ARAP-Zone-Access
  Description =
  Min = 0
  Max = 1
Auth-Grace-Period/
  Name = Auth-Grace-Period
  Description =
  Min = 0
  Max = 1
### Configuring Diameter Dictionary

The Diameter dictionary contains a list of application specific AVPs. Table 4-54 describes the Diameter BaseProtocol AVP Properties.

**Table 4-54 Diameter BaseProtocol AVP Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; name of the application specific AVPs.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional; description of the application specific AVPs.</td>
</tr>
<tr>
<td>IsVendorSpecific</td>
<td>Required; default is FALSE. If set to FALSE, the application is ordinary</td>
</tr>
<tr>
<td></td>
<td>application and user is prompted to enter ApplicationID. If set to TRUE,</td>
</tr>
<tr>
<td></td>
<td>the application is a Vendor Specific Application. User is prompted to enter</td>
</tr>
<tr>
<td></td>
<td>VendorSpecificApplicationID and VendorID.</td>
</tr>
<tr>
<td>ApplicationID</td>
<td>Required; specifies the unique integer value for the application.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> The Application ID must be set to 0 for BaseProtocol AVPs.</td>
</tr>
<tr>
<td>VendorSpecificApplicationID</td>
<td>Required, Specifies the integer value for the vendor specific application.</td>
</tr>
<tr>
<td>VendorID</td>
<td>Required, specifies the VendorID for the application.</td>
</tr>
<tr>
<td>AVPS/</td>
<td>Specifies the list of application specific avps.</td>
</tr>
<tr>
<td></td>
<td><strong>Example:</strong></td>
</tr>
<tr>
<td></td>
<td>Accounting-Realtime-Required/</td>
</tr>
<tr>
<td></td>
<td>Name = Accounting-Realtime-Required</td>
</tr>
<tr>
<td></td>
<td>Description =</td>
</tr>
<tr>
<td></td>
<td>Attribute = 483</td>
</tr>
<tr>
<td></td>
<td>Mandatory = Must</td>
</tr>
<tr>
<td></td>
<td>May-Encrypt = Yes</td>
</tr>
<tr>
<td></td>
<td>Protected = MustNot</td>
</tr>
<tr>
<td></td>
<td>Type = UINT32</td>
</tr>
<tr>
<td></td>
<td>Min = 0</td>
</tr>
<tr>
<td></td>
<td>Max = 253</td>
</tr>
</tbody>
</table>

Refer to Table 4-55 for the description of AVP properties.
Table 4-55 lists the application specific AVP properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>Specifies the integer value for the AVP.</td>
</tr>
<tr>
<td>Mandatory</td>
<td>Specifies whether the mandatory bit of this AVP should or should not be set.</td>
</tr>
<tr>
<td>May-Encrypt</td>
<td>If set to 'yes' then the AVP will be sent encrypted if the connection uses CMS security.</td>
</tr>
<tr>
<td>Protected</td>
<td>Specifies whether the protected bit of this AVP should or should not be set.</td>
</tr>
<tr>
<td>Type</td>
<td>Specifies the type of the Diameter AVP.</td>
</tr>
</tbody>
</table>

**Configure the Diameter Dictionary**

To configure the Diameter Dictionary:

**Step 1** Change to /Radius/Advanced/Diameter/Diameter Dictionary.

**Step 2** Add BaseProtocolAVPs.

```
add BaseProtocolAVPs
```

```
[ //localhost/Radius/Advanced/Diameter/Diameter Dictionary ]
```

```
cd BaseProtocolAVPs/
```

**Step 3** Set the properties for BaseProtocolAVPs.

```
[ //localhost/Radius/Advanced/Diameter/Diameter Dictionary/BaseProtocolAVPs ]
Name = BaseProtocolAVPs
Description =
IsVendorSpecific = FALSE
ApplicationID = 0
AVPs/
```

```
set IsVendorSpecific "FALSE"
```

```
set IsVendorSpecific "FALSE"
```

```
set ApplicationID 0
```

```
set ApplicationID 0
```

**Step 4** Configure the application specific AVPs.

```
cd AVPs/
```

```
add User-Name
```
Step 5  Configure User-Name AVP type and number

--> cd User-Name/

[ //localhost/Radius/Advanced/Diameter/Diameter Dictionary/BaseProtocolAVPs/AVPs/User-Name ]

  Name = User-Name
  Description =
  Attribute = 1
  Mandatory = MustNot
  May-Encrypt = No
  Protected = MustNot
  Type = UTF8_STRING
  Min = 0
  Max = 253

  set Attribute 1

  set Attribute  1

  set Type UTF8_STRING

  set Type UTF8_STRING

The following is an example of Diameter BaseProtocol AVPs:

[ //localhost/Radius/Advanced/Diameter/Diameter Dictionary/BaseProtocolAVPs ]

  Name = BaseProtocolAVPs
  Description =
  IsVendorSpecific = FALSE
  ApplicationID = 0
  AVPs/
      Entries 1 to 55 from 55 total entries
      Current filter: <all>

      Accounting-Realtime-Required/
          Name = Accounting-Realtime-Required
          Description =
          Attribute = 483
          Mandatory = Must
          May-Encrypt = Yes
          Protected = MustNot
          Type = UINT32
          Min = 0
          Max = 253

      Accounting-Record-Number/
          Name = Accounting-Record-Number
          Description =
          Attribute = 485
          Mandatory = Must
          May-Encrypt = Yes
          Protected = MustNot
          Type = UINT32
          Min = 0
          Max = 253

      Accounting-Record-Type/
          Name = Accounting-Record-Type
          Description =
         Attribute = 480
          Mandatory = Must
          May-Encrypt = Yes
          Protected = MustNot
          Type = ENUM
Min = 0
Max = 253

Enums/
    1 = "Event Record"
    2 = "Start Record"
    3 = "Interim Record"
    4 = "Stop Record"

Accounting-Session-Id/
    Name = Accounting-Session-Id
    Description =
    Attribute = 44
    Mandatory = Must
    May-Encrypt = Yes
    Protected = May
    Type = STRING
    Min = 0
    Max = 253

Accounting-Sub-Session-Id/
    Name = Accounting-Sub-Session-Id
    Description =
    Attribute = 287
    Mandatory = Must
    May-Encrypt = Yes
    Protected = May
    Type = UINT64
    Min = 0
    Max = 253
Using the radclient Command

This chapter describes how to use radclient, a RADIUS server test tool you run from the command line to test your Cisco Prime Access Registrar RADIUS server. You can use radclient to create packets, send them to a specific server, and examine the response.

Because the radclient command is Tcl-based, you can use it interactively or you can execute it as a Tcl script file.

To run the radclient command, enter:

```
radclient
```

After you enter the radclient command, you must log into the RADIUS server and provide an administrator’s username, such as admin, and the administrator’s password.

This chapter contains the following sections:

- radclient Command Syntax
- Working with Packets
- Attributes
- Using radclient Test Commands

radclient Command Syntax

The radclient command syntax is:

```
```

Valid flags are:

- `-C <clusternamename>`
- `-N <adminname>`
- `-P <adminpassword>`
- `-i`—Forces interactive mode
- `-n`—Skips loading radclient.tcl
- `-p <path>`—Specifies the load_path
- `-s`—Uses default cluster, admin user, and password

If you delete the admin user or modify the admin user’s password, this option will no longer work.
Working with Packets

Using the `radclient` command, you can create packets (default or specific packets), view packets, send packets, read the value of packets, and delete packets.

This section contains the following topics:

- Creating Packets
- Creating CHAP Access-Request Packets
- Viewing Packets
- Sending Packets
- Creating Empty Packets
- Setting Packet Fields
- Reading Packet Fields
- Deleting Packets

Creating Packets

To create a basic RADIUS Access-Request packet, use the `radclient` command `simple`. This function creates a packet and fills in basic attributes. The syntax of the `simple` command is:

```
simple <user_name> <user_password>
```

For example, to create an Access-Request packet for user `bob` whose password is `bigDog`, enter:

```
simple bob bigDog
```

`p001`

The `radclient` command responds with `p001`, which is the identifier (name) of the newly created packet.
Creating CHAP Access-Request Packets

To create a CHAP Access-Request packet, use the `radclient` command `simple_chap`. The syntax of the `simple_chap` command is:

```
simple_chap <user_name> <user_password> <use_challenge>
```

`<use_challenge>` is a boolean that indicates whether to use the CHAP-Challenge attribute.

For example, to create a CHAP packet and use a `<use_challenge>`, enter:

```
simple_chap bob bigDog 1
```

Viewing Packets

To view a packet or any other object, enter the object identifier at the `radclient` prompt. For example, to display packet `p001`, enter:

```
p001
```

Packet: code=Access-Request, id=0, length=0, attributes =
- User-Name = bob
- User-Password = bigDog
- NAS-Identifier = localhost
- NAS-Port = 0

Sending Packets

To send a packet, specify the packet identifier and enter the word `send`.

```
p001 send
```

You can optionally specify the host and port to which to send the packet. The default host is `localhost`, and the default port is `1812`.

When you want to send a packet to a different host and different port, you must specify them on the command line. For example, to send a packet to the RADIUS server `amazon`, at port number `1813`, enter:

```
p001 send amazon 1813
```

When Prime Access Registrar receives a response to the packet you sent, it prints the response packet’s object identifier before the `radclient` prompt returns.

The TCL variable `tries` determines how many times `radclient` retries to send the packet.
Creating Empty Packets

You can use `radclient` to create empty packets, then modify the packets to contain the appropriate fields. To create an empty packet, the syntax is:

```
packet <packet-type>
```

The optional `<packet-type>` argument can be the numerical RADIUS packet type identifier, such as 2, or the string representation, such as `Access-Accept`:

```
packet 2
```

```
p00d
```

```
p00d
```

Packet: code = Access-Accept, id = 0, length = 0, attributes =

Setting Packet Fields

You can modify the value of a packet field using the following syntax:

```
<packet-identifier> set <field> <value>
```

`<packet-identifier>` is the packet number, such as `p001`.

`<field>` is the packet field you want to modify and can be one of the following:

- `attrib`—Set attributes in the packet; `<value>` is the attribute identifier.
- `code`—The packet type (such as `Access-Request`); `<value>` is either a numeric packet-type or the string representation (for example, 1 or `Access Request`).
- `identifier`—Set the packet ID; `<value>` is the numeric ID.
- `length`—Set the packet length; `<value>` is the numeric length.
- `requestAuthenticator`—Set the request authenticator; `<value>` is a hex string with a colon separating each byte.

`<value>` is either a numeric packet-type, the string representation, or the hex string with a colon separating each byte.

For example, to set the identifier field to 99, enter:

```
p001 set identifier 99
```

```
99
```

```
p001
```

Packet: code = Access-Request, id = 99, length = 0, attributes =

```
User-Name = bob
User-Password = bigDog
NAS-Identifier = localhost
NAS-Port = 0
```
Reading Packet Fields

You can read (get) the value of any of the packet fields by using the syntax:

\[ <\text{packet-identifier}> \text{ get } <\text{attrib}> \]

For example, to get the identifier field, enter:

\[ p001 \text{ get identifier} \]

99

Deleting Packets

When you are writing long-running or iterating scripts, you might want to conserve memory by deleting packets when you are finished with them.

To delete a packet, enter:

\[ <\text{packet-identifier}> \text{ delete} \]

To delete all resources referred to by the packet p001, enter:

\[ p001 \text{ delete} \]

Attributes

Using the \text{radclient} command you can create specific RFC-defined attributes of requests and responses.

This section contains the following topics:

- Creating Attributes
- Setting Multivalued Attributes
- Viewing Attributes
- Getting Attribute Information
- Deleting Attributes
- Using the radclient Command

Creating Attributes

To create an attribute object, the syntax is:

\[ <\text{attrib}> \text{ name } <\text{value}> \]

\(<\text{attrib}>\) is a recognized RADIUS attribute name. \(<\text{value}>\) is the value of the attribute.

For example, to create the attribute User-Name and set its value to bob, enter:

\[ \text{ attrib User-Name bob} \]

\[ a001 \]
Setting Multivalued Attributes

Prime Access Registrar supports setting multivalued attributes (MVAs) in radclient. Use the set mattrib command to set multivalued attributes, as shown in the following example:

```
simple bob bob
p001
attrib cisco-avpair blah
a005
attrib cisco-avpair boo
a006
p001 set mattrib a005
p001
Packet: code = Access-Request, id = 0, length = 0, attributes =
User-Name = bob
User-Password = bob
NAS-Identifier = localhost
NAS-Port = 1
Cisco-AVPair = blah

p001 set mattrib a006
p001
Packet: code = Access-Request, id = 0, length = 0, attributes =
User-Name = bob
User-Password = bob
NAS-Identifier = localhost
NAS-Port = 1
Cisco-AVPair = blah
Cisco-AVPair = boo
```

Viewing Attributes

To view an attribute, or any other object, type the object identifier at the radclient prompt. For example, to display attribute a001 created in the example above, enter:

```
a001
User-Name = bob
```
Getting Attribute Information

You can get the name and value of an attribute in various formats:

- get name—gets the name as a string
- get value—gets the value as a string
- get type—gets the name as an integer
- get valueAsInt—gets the value as an integer
- get valueAsIPAddress—gets the value as an IP address.

The following examples show how to get an attribute’s name, type, value, and value as integer:

```bash
a001 get name
User-Name

a001 get type
1

a001 get value
bob

a001 get valueAsInt
a001: the value is not an int
```

Deleting Attributes

When you are writing long running or iterating scripts, you might want to conserve memory by deleting attributes when you are finished with them (be sure not to delete attributes being referred to by other objects, like packets.)

To delete all resources referred to by the attribute `a001`, enter:

```
a001 delete
```

Using the radclient Command

The following three examples show how to use `radclient` to create, send, and modify packets.

Example 1

This example creates an Access-Request packet for user `jane` with password `jane`, and sends it to the default RADIUS server (`localhost`).

```
simple jane jane
```
The command `simple jane jane` creates the packet; the packet object identifier is `p001`. When you enter the packet object identifier, `radclient` displays the contents of the packet.

```
p001
```

```
Packet: code = Access-Request, id = 0, length = 0, attributes =
  User-Name = jane
  User-Password = jane
  NAS-Identifier = localhost
  NAS-Port = 0
```

When you enter the packet identifier and the command `send`, `radclient` sends the packet to the RADIUS server and prints the response packet object identifier.

```
p001 send
```

```
p002
```

```
Packet: code = Access-Accept, id = 1, length = 38, attributes =
  Login-IP-Host = 204.253.96.3
  Login-Service = Telnet
  Login-TCP-Port = 541
```

### Example 2

The following example creates a simple Access-Request packet, then adds other attributes to it.

```
simple jane jane
```

```
p003
```

```
Packet: code = Access-Request, id = 0, length = 0, attributes =
  User-Name = jane
  User-Password = jane
  NAS-Identifier = localhost
  NAS-Port = 0
```

The command line above shows creation of the packet `p003` using user-ID `jane` and password `jane`

```
attrib Service-Type Framed
```

```
a00c
```

```
Service-Type = Framed
```

Entering the attribute object identifier `a00c` displays the attribute object.

```
p003 set attrib a00c
```

```
The line above adds the newly set attribute to the packet. The following line creates another attribute.
```

```
attrib NAS-Port 99
```
a00d

NAS-Port = 99

**p003 set attrib a00d**

The same steps add the **NAS-Port** attribute to the packet, and finally, the packet contents are displayed.

**p003**

Packet: code = Access-Request, id = 0, length = 0, attributes =

User-Name = jane
User-Password = jane
NAS-Identifier = localhost
Service-Type = Framed
NAS-Port = 99

**Example 3**

Example 3 performs the same tasks as **Example 2** using the command substitution feature of Tcl which allows you to use the results of one command as an argument to another command. Square brackets invoke command substitution. The statement inside the brackets is evaluated, and the result is used in place of the bracketed command.

**simple jane jane**

**p004**

**p004 set attrib [ attrib Service-Type Framed ]**

**p004 set attrib [ attrib NAS-Port 99 ]**

**p004**

Packet: code = Access-Request, id = 0, length = 0, attributes =

User-Name = jane
User-Password = jane
NAS-Identifier = localhost
Service-Type = Framed
NAS-Port = 99
Using radclient Test Commands

You can use the `radclient` commands `timetest` and `callsPerSecond` to test the RADIUS server.

This section contains the following topics:
- radclient Variables
- Using `timetest`
- Using `callsPerSecond`
- Additional radclient Variables

radclient Variables

You control how `timetest` and `callsPerSecond` work using `radclient` variables. To set a `radclient` variable, use the `set` command as follows:

```
set variable value
```

Table 5-1 lists the `radclient` variables used in `timetest` and `callsPerSecond` and their description.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Destination host to send the packets (default is localhost)</td>
</tr>
<tr>
<td>num_packets</td>
<td>Number of packets to send at once (default is 256)</td>
</tr>
<tr>
<td>num_users</td>
<td>Modulus for the username pattern (default is 10000)</td>
</tr>
<tr>
<td>port</td>
<td>Port where <code>radclient</code> sends access-request packets (default is 1812). Changing this port does not affect the accounting_port.</td>
</tr>
<tr>
<td>retry_timeout</td>
<td>Length of time to wait after a timeout occurs before retrying</td>
</tr>
<tr>
<td>secret</td>
<td>Shared secret configured on the RADIUS server for the client (default is secret)</td>
</tr>
<tr>
<td>timeout</td>
<td>Length of time to wait before a timeout occurs</td>
</tr>
<tr>
<td>tries</td>
<td>Number of times to attempt to send</td>
</tr>
<tr>
<td>UserNamePattern</td>
<td>Pattern of the usernames (default is user%d%%PPP)</td>
</tr>
<tr>
<td>UserPasswordPattern</td>
<td>Pattern of the user passwords (default is user%d)</td>
</tr>
</tbody>
</table>

Using timetest

The `timetest` command sends a number of requests to the RADIUS server then waits for a response. When a response arrives, `timetest` immediately sends another request. `timetest` can keep up to 256 requests outstanding all the time.

The syntax of the `timetest` command is:

```
timetest <testtype> [<cycles> [<repetitions> [<starting user number> [<increment user number>]]]]
```

Table 5-2 lists the applicable test types.
Consider this `timetest` example with `radclient` variables set to the following:

- host—1.1.1.2
- port—1812
- secret—cisco
- UserNamePattern—user%d
- UserPasswordPattern—puser%d
- num_users—100,000
- num_packets—128

In this example, `timetest` sends packets directly to the host at IP address 1.1.1.2 on port 1812 with a shared secret `cisco`. There are 100,000 users in the server’s user database with the name pattern `user#` and password pattern `puser#`, where # ranges from 0-99,999, inclusive. The number of outstanding requests are limited to 128.

Before starting the timing test, `timetest` sends an Accounting-On packet to the AAA Server and waits for a response to make sure that any session management being performed on the AAA Server is reset before running the test. After a response is received, the `timetest` can begin.

### Using `callsPerSecond`

The `callsPerSecond` command is a smart throttle that sends packets at a rate you set. If you set `callsPerSecond` to two transactions per second (TPS), `callsPerSecond` sends a packet every 0.5 seconds.

The syntax of the `callsPerSecond` command is:

```
callsPerSecond <callsPerSecond> <testtype> [<cycles> [<repetitions> [<starting user number> [<increment user number>]]]]
```

### Table 5-2: Test Types

<table>
<thead>
<tr>
<th>Test Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Access-Request</td>
</tr>
<tr>
<td>2</td>
<td>Access-Request + Accounting-Start + Accounting-Stop</td>
</tr>
<tr>
<td>3</td>
<td>Accounting-Start + Accounting-Stop</td>
</tr>
<tr>
<td>4</td>
<td>Ascend-IPA-Allocate + Ascend-IPA-Release</td>
</tr>
<tr>
<td>5</td>
<td>Access-Request + Ascend-IPA-Allocate + Ascend-IPA-Release</td>
</tr>
<tr>
<td>6</td>
<td>Access-Request + Ascend-IPA-Allocate + Accounting-Start + Ascend-IPA-Release + Accounting-Stop</td>
</tr>
<tr>
<td>7</td>
<td>Access-Request + USR-Resource-Free-Request</td>
</tr>
<tr>
<td>8</td>
<td>LEAP Identity + LEAP-Challenge Response + LEAP Challenge</td>
</tr>
<tr>
<td>9</td>
<td>LEAP Identity + LEAP-Challenge Response + LEAP Challenge + Accounting-Start + Accounting-Stop</td>
</tr>
<tr>
<td>10</td>
<td>Access-Request + Accounting-Start + Accounting-Stop with Home-Agent request</td>
</tr>
<tr>
<td>11</td>
<td>Access-Request + Accounting-Start + Accounting-Stop with ODAP request</td>
</tr>
</tbody>
</table>
## Additional radclient Variables

Table 5-3 lists additional **radclient** variables and their description.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accounting_port</td>
<td>Port where the RADIUS server sends accounting packets (default is 1813).</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> Changing accounting_port value does not affect the authentication port.</td>
</tr>
<tr>
<td>host</td>
<td>Name of host where Prime Access Registrar is installed</td>
</tr>
<tr>
<td>ignore_signature_errs</td>
<td>Causes server to ignore signature in the response</td>
</tr>
<tr>
<td>load_path</td>
<td>Search path to load source files with user processes</td>
</tr>
<tr>
<td>NASIdentifier</td>
<td>Value to set NAS-Identifier attribute</td>
</tr>
<tr>
<td>NASIPAddress</td>
<td>Value to set NAS-IP-Address attribute</td>
</tr>
<tr>
<td>NASPort</td>
<td>Value to set NAS-Port attribute</td>
</tr>
<tr>
<td>num_packets</td>
<td>Number of packets to send at once (default is 256)</td>
</tr>
<tr>
<td>num_users</td>
<td>Modulus for the username pattern (default is 10000)</td>
</tr>
<tr>
<td>port</td>
<td>Port where <strong>radclient</strong> sends access-request packets (default is 1812).</td>
</tr>
<tr>
<td></td>
<td>Changing this port does not affect the accounting_port.</td>
</tr>
<tr>
<td>retry_timeout</td>
<td>Length of time to wait before attempting a retry</td>
</tr>
<tr>
<td>secret</td>
<td>Shared secret configured on the RADIUS server for the client (default is secret)</td>
</tr>
<tr>
<td>tclDefaultLibrary</td>
<td>Tclsh default library</td>
</tr>
<tr>
<td>tcl_patchLevel</td>
<td>Tclsh version with patch level</td>
</tr>
<tr>
<td>tcl_pkgPath</td>
<td>Tclsh install path</td>
</tr>
<tr>
<td>tcl_traceExec</td>
<td>Tclsh boolean to activate tracing</td>
</tr>
<tr>
<td>tcl_platform</td>
<td>Tclsh platform array</td>
</tr>
<tr>
<td>tcl_version</td>
<td>Tclsh version</td>
</tr>
<tr>
<td>tries</td>
<td>Number of retry attempts</td>
</tr>
<tr>
<td>UserNamePattern</td>
<td>Pattern of the usernames (default is user%d%%PPP)</td>
</tr>
<tr>
<td>UserPasswordPattern</td>
<td>Pattern of the user passwords (default is user%d)</td>
</tr>
<tr>
<td>verbose</td>
<td>Verbose flag for Tclsh</td>
</tr>
</tbody>
</table>
CHAPTER 6

Configuring Local Authentication and Authorization

Cisco Prime Access Registrar (Prime Access Registrar) allows user information to be stored in its own internal database or external stores such as an LDAP directory or Oracle database. This chapter describes how to configure Prime Access Registrar to perform authentication and authorization using the Prime Access Registrar internal database and how to verify and troubleshoot a local service and userlist configuration.

In RADIUS, an Access Request packet is a request for authentication and authorization (AA). Authentication checks username and password credentials, while authorization typically involves returning the correct information to allow the service a user is authorized to have. Prime Access Registrar performs AA and returns the appropriate RADIUS attributes in an Access Accept packet.

This chapter contains the following sections:

- Configuring a Local Service and UserList
- Troubleshooting the Local Service and UserList Configuration
- aregcmd Command Performance
- UserDefined1 Property
- Access-Request Logging

## Configuring a Local Service and UserList

Prime Access Registrar uses services configured under /Radius/Services to process RADIUS requests. To process RADIUS access requests locally, you must configure a service and set its type to local. A local service references an Prime Access Registrar userlist.

The following sections show the commands you enter and the expected responses from the Prime Access Registrar server to do the following:

- Configuring a Local Service
- Configuring a Userlist
- Configuring Cisco Prime Access Registrar to Use the Local Service For AA
- Activating the Configuration
Throughout this chapter, the `aregcmd` commands you enter are shown in **bold** font, and the server responses are shown in *smaller plain font* as shown in the following:

```
command you enter

server response
```

## Configuring a Local Service

Prime Access Registrar maintains **Services** under `/Radius`. To configure a local service:

**Step 1** Use the `add` command at `/Radius/Services` to create a Service.

```
   cd /Radius/Services
   add SouthBay
```

```
   Added SouthBay
```

**Step 2** Change directory to the new service and set its type to local.

```
   cd SouthBay
```

```
   [ //localhost/Radius/Services/SouthBay ]
   set type local
```

```
   Set Type local
```

**Step 3** Use the `set` command to associate a userlist with the service.

```
   set userlist SouthUsers
```

```
   Set UserList SouthUsers
```
Chapter 6 Configuring Local Authentication and Authorization

Configuring a Userlist

Prime Access Registrar maintains UserLists under /Radius.
To configure a userlist:

Step 1 Use the add command at /Radius/UserLists to create a userlist.

```bash
cd /Radius/UserLists
[ //localhost/Radius/UserLists ]
add SouthUsers
Added SouthUsers
```

Step 2 Change directory to the userlist and add users.

```bash
cd SouthUsers
[ //localhost/Radius/UserLists/SouthUsers ]
add user1
Added user1
```

Step 3 Change directory to each user you add and set the user’s password.

```bash
cd user1
[ //localhost/Radius/UserLists/SouthUsers/user1 ]
set Password test
Retype password to confirm:
Set Password <encrypted>
```

Configuring Cisco Prime Access Registrar to Use the Local Service For AA

To configure Prime Access Registrar to use the local service for authentication and authorization, enter commands to set the DefaultAuthenticationService and DefaultAuthenticationService to the service you created, as shown in the following:

```bash
cd /Radius
[ //localhost/Radius ]
```
Chapter 6  Configuring Local Authentication and Authorization

Activating the Configuration

To activate the configuration changes you have made, enter the save command:

```
set DefaultAuthenticationService SouthBay
Set DefaultAuthenticationService SouthBay
set DefaultAuthorizationService SouthBay
Set DefaultAuthorizationService SouthBay
```

```
save
Validating //localhost...
Saving //localhost...
```

After you issue the save command, Prime Access Registrar attempts to validate the configuration, checks for all required properties, and ensures there are no logic errors. If the validation is successful, Prime Access Registrar saves the configuration to the MCD database.

Troubleshooting the Local Service and UserList Configuration

Before you begin troubleshooting, ensure that the current configuration is valid and active. To ensure that any configuration changes you have made are valid and stored in the database, you must issue the save command.

```
save
Validating //localhost...
Saving //localhost...
```

To ensure that the current configuration is active, issue the reload command.

```
reload
Reloading Server 'Radius'...
Server 'Radius' is Running, its health is 10 out of 10
```

Verifying the Configuration

To verify the configuration changes you have made:

**Step 1** Check to see that the UserList exists under the service.

```
ls /Radius/Services/SouthBay
```
Chapter 6  Configuring Local Authentication and Authorization

Troubleshooting the Local Service and UserList Configuration

[ /Radius/Services/SouthBay ]
Name = SouthBay
Description =
Type = local
IncomingScript~ =
OutgoingScript~ =
OutagePolicy~ = RejectAll
OutageScript~ =
UserList = SouthUsers

Step 2  Check to see that user user1 exists under the SouthUsers userlist.

ls /Radius/UserLists/SouthUsers

[ /Radius/UserLists/SouthUsers ]
Entries 1 to 1 from 1 total entries
Current filter: <all>
Name = SouthUsers
Description =
user1/

Step 3  Turn on debugging.

trace /r 5

Traced "/Radius: Trace level is set to 5"

Step 4  Use radclient to send an Access-Request for user user1.

simple user1 test

The debugging output will be sent to the file name_radius_1_log in the /opt/CSCOar/logs directory. The following example shows items you should expect in a successful Access-Request.

Note  Lines of interest are in bold font.

11/12/2013 18:34:35: P1144: Packet received from 127.0.0.1
11/12/2013 18:34:35: P1144: Trace of Access-Request packet
11/12/2013 18:34:35: P1144: Identifier = 4
11/12/2013 18:34:35: P1144: length = 62
11/12/2013 18:34:35: P1144: User-Name = user1
11/12/2013 18:34:35: P1144: NAS-Port = 2
11/12/2013 18:34:35: P1144: NAS-Identifier = localhost
11/12/2013 18:34:35: P1144: Using Client: localhost (127.0.0.1)
11/12/2013 18:34:35: P1144: Using NAS: localhost (127.0.0.1)
11/12/2013 18:34:35: P1144: Request is directly from a NAS: TRUE
11/12/2013 18:34:35: P1144: Authenticating and Authorizing with Service SouthBay
11/12/2013 18:34:35: P1144: Getting user1's UserRecord from UserList SouthUsers
11/12/2013 18:34:35: P1144: User user1's password matches
11/12/2013 18:34:35: P1144: Trace of Access-Accept packet
11/12/2013 18:34:35: P1144: Identifier = 4
11/12/2013 18:34:35: P1144: length = 20
11/12/2013 18:34:35: P1144: Sending response to 127.0.0.1
11/12/2013 18:34:35: Log: Request from localhost (127.0.0.1): User user1 accepted
The following example shows a trace for an unsuccessful Access-Request due to an invalid password.

Note Lines of interest are in bold font.

11/12/2013 19:05:13: P1527: Packet received from 127.0.0.1
11/12/2013 19:05:13: P1527: Using NAS: localhost (127.0.0.1)
11/12/2013 19:05:13: P1527: Request is directly from a NAS: TRUE
11/12/2013 19:05:13: P1527: Authenticating and Authorizing with Service SouthBay
11/12/2013 19:05:13: P1527: Getting User user1's UserRecord from UserList SouthUsers
11/12/2013 19:05:13: P1527: User user1's password does not match
11/12/2013 19:05:13: P1527: Rejecting request
11/12/2013 19:05:13: P1527: Rejecting request
11/12/2013 19:05:13: Log: Request from localhost (127.0.0.1): User user1 rejected (UserPasswordInvalid)

If a user’s password is invalid, reset the password to ensure it was entered correctly. Also check that the shared secret being used by the RADIUS client and the Prime Access Registrar server match.

### Configuring Return Attributes and Check-Items

Prime Access Registrar supports RADIUS check item attributes at the user and group levels. You can configure Prime Access Registrar to check for attributes that must be present or attributes that must not be present in the Access-Request packet for successful authentication. For a complete list of attributes supported in Prime Access Registrar, see Appendix C, “RADIUS Attributes”.

When using check item attributes, Prime Access Registrar rejects Access-Requests if either of the following conditions exist:

- Any configured check item attributes are not present in the Access-Request packet
- Any Access-Request packet’s check item attribute values do not match with those configured check item attribute values

This section contains the following topics:

- Configuring Per User Return Attributes
- Configuring Per User Check-Items
- Verifying the Per User Return Attributes and Check-Items Configuration
- Configuring Return Attributes and Check-Items Using UserGroup

### Configuring Per User Return Attributes

User return attributes are attributes that are specific for a given user each time they log in. To configure a user’s return attributes, change directory to the user’s Attributes subdirectory and configure the desired attributes.

```
cd /Radius/UserLists/SouthUsers/User1/Attributes
```
Chapter 6  Configuring Local Authentication and Authorization

Troubleshooting the Local Service and UserList Configuration

[ //localhost/Radius/UserLists/SouthUsers/user1/Attributes ]

```
set Session-Timeout 60
Set Session-Timeout 60
```

```
set Callback-Number 5551124
Set Callback-Number 5551124
```

Configuring Per User Check-Items

Check Items are a way to check that certain attribute/values exist in a user’s access-request. If the attribute/values are not present in the access-request, the Prime Access Registrar server rejects the access-request.

To check that an access-request for user1 has the Calling-Station-Id attribute set to 5555678, enter the following:

```
cd /Radius/UserLists/SouthUsers/User1/CheckItems
```

```
set Calling-Station-Id 5555678
Set Calling-Station-Id 5555678
```

Be sure to save your configuration to preserve your changes.

Verifying the Per User Return Attributes and Check-Items Configuration

A successful request will produce a trace similar to the following:

```
11/12/2013 14:08:07: P1539: Packet received from 127.0.0.1
11/12/2013 14:08:07: P1539: Trace of Access-Request packet
11/12/2013 14:08:07: P1539: Identifier = 1
11/12/2013 14:08:07: P1539: length = 71
11/12/2013 14:08:07: P1539: User-Name = user1
11/12/2013 14:08:07: P1539: NAS-Port = 1
11/12/2013 14:08:07: P1539: Calling-Station-Id = 5555678
11/12/2013 14:08:07: P1539: NAS-Identifier = localhost
11/12/2013 14:08:07: P1539: Using Client: localhost (127.0.0.1)
11/12/2013 14:08:07: P1539: Using NAS: localhost (127.0.0.1)
11/12/2013 14:08:07: P1539: Request is directly from a NAS: TRUE
11/12/2013 14:08:07: P1539: Authenticating and Authorizing with Service SouthBay
11/12/2013 14:08:07: P1539: Getting User user1's UserRecord from UserList SouthUsers
11/12/2013 14:08:07: P1539: User user1's password matches
11/12/2013 14:08:07: P1539: Processing User user1's check items
11/12/2013 14:08:07: P1539: Merging User user1's Attributes into response Dictionary
11/12/2013 14:08:07: P1539: Merging attributes into the Response Dictionary:
11/12/2013 14:08:07: P1539: Adding attribute Callback-Number, value = 5551124
11/12/2013 14:08:07: P1539: Adding attribute Session-Timeout, value = 60
```
Configuring Profiles to Group Attributes

You can use the Prime Access Registrar profile object to group attributes. For example, you might want to group attributes for all PPP users. All PPP users could then be assigned the profile and the attributes contained in the profile would be returned in their access-accepts.

To configure profiles to group attributes:

**Step 1** Change directory to `/Radius/Profiles` and add a profile.

```
cd /Radius/Profiles
```

[ //localhost/Radius/Profiles ]

```
add PPP-Profile
```

Added PPP-Profile

**Step 2** Change directory to the new profile, then change directory to the profile’s Attributes subdirectory.

```
cd PPP-Profile
```

[ //localhost/Radius/Profiles/PPP-Profile ]

```
cd Attributes
```

[ //localhost/Radius/Profiles/PPP-Profile/Attributes ]

**Step 3** Configure the desired attributes for the profile.

```
set Service-Type Framed
```

Set Service-Type Framed

```
set Framed-Protocol PPP
```

Set Framed-Protocol PPP
Note When you need to set an attribute to a value that includes a space, you must double-quote the value, as in the following: set Framed-Route "192.168.1.0/24 192.168.1.1"

Step 4 Assign the profile to a user by setting the user’s BaseProfile attribute to the desired profile.

    cd /Radius/UserLists/SouthUsers/User1

    set BaseProfile PPP-Profile

Set BaseProfile PPP-Profile

Configuring Return Attributes and Check-Items Using UserGroup

A profile can also be assigned to a UserGroup. You assign a profile to a group by setting the group’s BaseProfile attribute to the desired profile.

To configure return attributes and check-items using usergroup:

Step 1 Change directory to /Radius/UserGroups and add a UserGroup.

    cd /Radius/UserGroups

    add PPP-Group

Added PPP-Group

Step 2 Change directory to the new UserGroup and add Return Attributes.

    cd PPP-Group

    cd Attributes

    set Service-Type Outbound

Set Service-Type Outbound

Step 3 Change directory to the UserGroups’ Check-Items subdirectory and add CheckItems.
cd ..:/CheckItems/  
[ //localhost/Radius/UserGroups/PPP-Group/CheckItems ]

set Service-Type Framed  
Set Service-Type Framed

Step 4 Assign the UserGroup to a User.  

cd /Radius/UserLists/SouthUsers/User2  
[ //localhost/Radius/UserLists/SouthUsers/user2 ]

set Group PPP-Group  
Set Group PPP-Group

---

**Return Attribute Precedence**

Because there are multiple ways of returning attributes, you might at some time have an attribute clash. In case of an attribute clash, the attribute precedence is as follows (from highest to lowest):

1. User attribute  
2. User profile  
3. UserGroup attribute  
4. UserGroup profile

---

**aregcmd Command Performance**

You can impact aregcmd command performance and server response time by having Prime Access Registrar userlists that contain more than 10,000 users. Prime Access Registrar userlists were not designed to contain 10,000 users in any one list.

If you must provide service for groups greater than 10000 users, we recommend that you use an external data store such as an LDAP directory or an Oracle database. If you are unable to use an external data store, create multiple userlists instead, keeping each userlist under 10,000 users.

Multiple userlists require multiple services (one for each userlist), because a service cannot reference more than one userlist. The multiple services can then be combined using the Service Grouping feature with ResultRule, OR, as follows:

[ //localhost/Radius/Services/GroupService ]
Name = GroupService  
Description =  
Type = group  
IncomingScript~ =  
OutgoingScript~ =  
ResultRule = OR
UserDefined1 Property

The UserDefined1 property of a user object is a free text field. You can use the UserDefined1 property to store additional user information much like the Description property, but its most powerful use is to pass information to an extension point script. The value set in the UserDefined1 property is automatically set to the environment variable of the same name during authentication. Any extension point script that subsequently runs has access the value in that property.

```plaintext
[ //localhost/Radius/UserLists/Default/bob ]
   Name = bob
   Description =
   Password = <encrypted>
   AllowNullPassword = FALSE
   Enabled = TRUE
   Group- =
   BaseProfile- =
   AuthenticationScript- =
   AuthorizationScript- =
   UserDefined1 =
   Attributes/
   CheckItems/
```

Access-Request Logging

By default, Prime Access Registrar logs all dropped and rejected requests in the name_radius_1_log file. The following are examples of log entries for dropped or rejected requests.

11/12/2013 17:38:11 name/radius/1 Warning Protocol 0 Request from localhost (127.0.0.1): User user1 rejected (UserPasswordInvalid)

11/12/2013 18:05:12 name/radius/1 Warning Protocol 0 Packet from 128.107.132.106: that address is not in the Clients list <unknown user>

To log all accepted requests as well, set the LogServerActivity advanced property to TRUE:

```
set /Radius/Advanced/LogServerActivity TRUE
```

```
Set /Radius/Advanced/LogServerActivity TRUE
```

```
save
```

Validating //localhost...

Saving //localhost...
reload

Reloading Server 'Radius'...
Server 'Radius' is Running, its health is 10 out of 10

Access-Accept packets are now logged as well:
11/12/2013 18:22:32 name/radius/1 Activity Protocol 0 Request from localhost (127.0.0.1):
User user2 accepted
RADIUS Accounting

This chapter describes RADIUS Accounting in Cisco Prime Access Registrar (Prime Access Registrar) as defined in Internet RFC 2866.

This chapter contains the following sections:

- Understanding RADIUS Accounting
- Setting Up Accounting
- Oracle Accounting
- LDAP Accounting
- MySQL Support
- Proxying Accounting Records
- Accounting Log Examples
- Sample Error Messages

Understanding RADIUS Accounting

RADIUS accounting is the process of collecting and storing the information contained in Accounting-Start and Accounting-Stop messages.

Internet RFC 2866 describes the protocol for sending accounting information between a Network Access Server (NAS) and a RADIUS server (or shared accounting server).

Note

Prime Access Registrar uses UDP port number 1813 as its default port for RADIUS accounting messages. RFC 2866 defines UDP port number 1813 as the accounting port number.

When a NAS that uses accounting begins a session, it sends an Accounting-Start packet describing the type of service and the user being connected to the Prime Access Registrar server. When the session ends, the NAS sends the RADIUS server an Accounting Stop packet describing the type of service that was delivered. The Accounting Stop packet might also contain statistics such as elapsed time, input and output octets, or input and output packets.
Setting Up Accounting

To configure Prime Access Registrar to perform accounting, you must do the following:

1. Create a service
2. Set the service type to file
3. Set the DefaultAccountingService field in /Radius to the name of the service you created

After you save and reload the Prime Access Registrar server configuration, the Prime Access Registrar server writes accounting messages to the accounting.log file in the /opt/CSCOar/logs directory. The Prime Access Registrar server stores information in the accounting.log file until a rollover event occurs. A rollover event is caused by the accounting.log file exceeding a pre-set size, a period of time transpiring, or on a scheduled date.

Note
You can also choose to export the accounting messages to a .csv file by providing the appropriate file type in the accounting service.

When the rollover event occurs, the data in accounting.log is stored in a file named by the prefix accounting, a date stamp (yyyymmdd), and the number of rollovers for that day. For example, accounting-20131107-14 would be the 14th rollover on November 07, 2013.

The following shows the properties for a service called CiscoAccounting:

```
[ //localhost/Radius/Services/acc ]
Name = acc
Description =
Type = file
IncomingScript~ =
OutgoingScript~ =
OutagePolicy~ = RejectAll
OutageScript~ =
FilenamePrefix = accounting
FileType~ = log
EnableRollOverIntelligence = TRUE
MaxFileSize = "10 Megabytes"
MaxFileAge = "1 Day"
RolloverSchedule =
UseLocalTimeZone = FALSE
AttributesToBeLogged:/
  1. Acct-Session-Id
```

Accounting Log File Rollover

The Prime Access Registrar accounting functionality provides flexibility in managing the accounting log. You can configure the Prime Access Registrar server to rollover the accounting log using any combination of the following Prime Access Registrar accounting service properties:

- MaxFileSize—Indicates the maximum size of the accounting log file in KB, MB, or GB
- MaxFileAge—Indicates the maximum age of the log file in minutes, hours, days, or weeks
- RolloverSchedule—Indicates the exact time including the day of the month or day of the week, hour and minute to roll over the accounting log file
You can configure an accounting service using any combination of MaxFileSize, MaxFileAge, and RolloverSchedule. For example, you might configure RolloverSchedule and MaxFileAge at the same time. This would be useful if you wanted to have an age-based rollover, but also synchronize to an absolute clock at specified times. The following would set a rollover every twelve hours at 11:59 and 12:59.

```
set MaxFileAge "12 H"
set RolloverSchedule "59 11,12 * * *"
```

You might also consider scheduling MaxFileAge to be six minutes and set RolloverSchedule to the top of the hour. The following would create ten six-minute long files starting anew every hour.

```
set MaxFileAge "6 Minutes"
set RolloverSchedule "0 * * * *"
```

Although you specify an exact time with the RolloverSchedule property, the Prime Access Registrar server only checks the rollover schedule when an accounting event occurs. If your Prime Access Registrar server receives a steady flow of packets (at least one per minute), the times you specify are accurate. However, if the Prime Access Registrar server does not receive any packets for a period of time, no rollovers will occur until the next packet is received. The same is true for MaxFileAge and MaxFileSize.

Based on the maximum file size and the age specified, Prime Access Registrar closes the accounting file, moves it to a new name, and reopens the file as a new file. The name given to this accounting file depends on its creation and modification dates.

For example, if the file was created and modified on the same date, the filename will be of the format `FileNamePrefix-<yyyymmdd>-<n>.log`, and the suffix will have year, month, day, and number. If the file was created on some day and modified on another, the filename will be of the format `FileNamePrefix-<yyyymmdd>-<yyyymmdd>-<n>.log`, and the suffix will have creation date, modification date, and number.

This section contains the following topics:

- FilenamePrefix
- MaxFileSize
- MaxFileAge
- RolloverSchedule
- UseLocalTimeZone
- FileType
- EnableRolloverIntelligence
- AttributesToBeLogged

### FilenamePrefix

The FileNamePrefix property enables you to specify a path to the file system in which you store the log files. If you do not manage your log files regularly, they might use the system resources, which will affect the performance of the Prime Access Registrar server.
We recommend that you store the log files in a file system different from the file system where you installed the Prime Access Registrar software by specifying the path in the FilenamePrefix property. By doing so the Prime Access Registrar server continues to run, even if the accounting logs fill the file system.

The following example specifies the /usr/arlogs/accounting as the FilenamePrefix:

```
set /Radius/Services/CiscoAccounting/FilenamePrefix /usr/arlogs/accounting
```

You can also set up a cron job to check the size of the log files and mail the administrator if the file system is full.

### MaxFileSize

Use MaxFileSize to indicate the maximum size of the accounting.log file in minutes, hours, days, or weeks. MaxFileAge measures the age of the accounting.log file from the time the previous file rollover occurred.

You can specify the following (case insensitive) file sizes:

- K, Kilobytes, Kilobytes
- M, Megabyte, Megabytes
- G, Gigabyte, Gigabytes

The following are examples of valid commands to set MaxFileSize:

```
set MaxFileSize “500 kilobytes”
```

The example above sets a MaxFileSize of 500 kilobytes

```
set maxfilesize “1 G”
```

The example above sets a MaxFileSize of one gigabyte

```
set maxfilesize “200 megabyte”
```

The example above sets a MaxFileSize of 200 megabytes

### MaxFileAge

Use MaxFileAge to indicate the maximum age of the log file in minutes, hours, days, or weeks. MaxFileAge measures the age of the accounting.log file from the time the previous file rollover occurred.

You can specify the following (case insensitive) periods of time:

- M, Minute, or Minutes preceded by a number from 0 to 59
- H, Hour, or Hours preceded by a number from 0 to 12
- D, Day, or Days preceded by a number from 1 to 31
- W, Week, or Weeks preceded by a number from 1 to 52

The following are examples of valid commands to set MaxFileAge:

```
set MaxFileAge “6 Minutes”
```

The example above sets a MaxFileAge of 6 minutes.
set maxfileage "2 d"

The example above sets a MaxFileAge of two days.

set maxfileage "1 H"

The example above sets a MaxFileAge of one hour.

**RolloverSchedule**

You set RolloverSchedule using the following crontab-style time format:

```
minute hour "day of month" "month of year" "day of week"
```

Where:

- Minute is a value from 0-59
- Hour is a value from 0-12
- Day (of the month) is a value from 1-31
- Month is a value from 1-12
- Day (of the week) is a value from 0-6, where 0 is Sunday

**UseLocalTimeZone**

When set to TRUE, the Prime Access Registrar server stores the accounting records in the log using the local system time. When set to FALSE (the default), Prime Access Registrar stores the accounting records in the log using Greenwich Mean Time (GMT).

**FileType**

Use **FileType** to indicate the type of the file to export the accounting messages to. FileType could be one of the following:

- **log**— Prime Access Registrar server writes accounting messages to the accounting.log file in the /opt/CSCOar/logs directory.
- **csv**— Prime Access Registrar server writes accounting messages to the accounting.csv file in the /opt/CSCOar/logs directory. You must set up a delimiter for this file type, which could be ‘;’, ‘,’ and ‘:’.

**EnableRolloverIntelligence**

When set to **TRUE**, rollover intelligence will be enabled for the accounting records based on the accounting service properties. For example, if a log file is deleted, this parameter will indicate whether to create a log with the deleted index before continuing with new indexes or to ignore the deleted index and create log files from the last index available for that date.

For example, if:

- there are log files such as acct-1-1209-2015.log, acct-2-1209-2015.log, through acct-10-1209-2015.log for that date
- EnableRolloverIntelligence is set to TRUE
- acct-2-1209-2015.log is deleted

**AttributesToBeLogged**

The `AttributesToBeLogged` parameter allows you to configure the set of attributes that must be logged by the accounting file service for a particular packet. If this list is empty, the accounting file service logs all the attributes available for that particular packet.

**Oracle Accounting**

Previous releases of Prime Access Registrar supported accessing user data from an Oracle database using Open Database Connectivity (ODBC), but this feature was limited to performing authentication and authorization (AA). You could only write the accounting records to local file or proxy to another RADIUS server.

Prime Access Registrar supports writing accounting records into Oracle database enabling integration between billing systems and Oracle.

- Prime Access Registrar adds a new type of service and remote server called `odbc-accounting` that enables inserting accounting records into Oracle.
- You can write accounting records into Oracle by referring this service in `/Radius/DefaultAccountingService` or in the `Accounting-Service` environment variable.

There is no specified schema structure to use the Oracle accounting feature. You can use your own table design and configure insert statements using standard SQL in the Prime Access Registrar configuration. The Prime Access Registrar server executes the insert statements to write the accounting record into Oracle. This feature is similar to the existing ODBC feature which performs authentication and authorization.

To improve latency for writing accounting records into database, packet buffering can be used. This option is enabled using the `BufferAccountingPackets` property under the `odbc-accounting` remote server definition.

---

**Note**

Prime Access Registrar supports Oracle 10g client and 11g server.

**Note**

For more information about dynamic SQL feature, see Dynamic SQL Feature, page 7-11.

This section contains the following topics:

- Configuring Oracle Accounting
- Packet Buffering
- Dynamic SQL Feature
Configuring Oracle Accounting

To use the Oracle accounting feature,

- you must configure a service of type `odbc-accounting` under `/Radius/Services`.
- you must also configure at least one remote server of type `odbc-accounting` under `/Radius/RemoteServers`.

This section contains the following topics:
- ODBC-Accounting Service
- Configuring Oracle Accounting
- ODBC RemoteServers
- Configuration Examples
- Packet Buffering
- Dynamic SQL Feature

ODBC-Accounting Service

The following is an example of an ODBC-Accounting service:

```
[ //localhost/Radius/Services/oracle_accounting ]
Name = oracle_accounting
Description =
Type = odbc-accounting
IncomingScript~ =
OutgoingScript~ =
OutagePolicy~ = RejectAll
OutageScript~ =
MultipleServersPolicy = Failover
RemoteServers/
  1. accounting_server
```

ODBC RemoteServers

Create a remote server under `/Radius/RemoteServers`, and set its protocol to `odbc-accounting`. The following is an example of an ODBC-Accounting RemoteServer’s configuration:

```
[ //localhost/Radius/RemoteServers/accounting_server ]
Name = accounting_server
Description =
Protocol = odbc-accounting
ReactivateTimerInterval = 300000
Timeout = 15
DataSourceConnections = 8
ODBCDataSource =
KeepAliveTimerInterval = 0
BufferAccountingPackets = TRUE
MaximumBufferSize = "10 Megabytes"
NumberOfRetriesForBufferedPacket = 3
BackingStoreEnvironmentVariables =
UseLocalTimeZone = FALSE
AttributeList =
Delimiter =
SQLDefinition/
```
Table 7-1 describes the ODBC RemoteServer properties.

### Table 7-1 ODBC RemoteServer Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the remote server; this property is mandatory, and there is no default</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of server</td>
</tr>
<tr>
<td>Protocol</td>
<td>Must be set to odbc-accounting</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Mandatory time interval (in milliseconds) to activate an inactive server; defaults to 300000 ms.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Mandatory time interval (in seconds) to wait for SQL operation to complete; defaults to 15 seconds</td>
</tr>
<tr>
<td>DataSourceConnections</td>
<td>Mandatory number of connections to be established; defaults to 8</td>
</tr>
<tr>
<td>ODBCDataSource</td>
<td>Name of the ODBCDataSource to use and must refer to one entry in the list of ODBC datasources configured under /Radius/Advanced/ODBCDataSources. Mandatory; no default</td>
</tr>
<tr>
<td>KeepAliveTimerInterval</td>
<td>Mandatory time interval (in milliseconds) to send a keepalive to keep the idle connection active; defaults to zero (0) meaning the option is disabled</td>
</tr>
<tr>
<td>BufferAccountingPackets</td>
<td>Mandatory, TRUE or FALSE, determines whether to buffer the accounting packets to local file, defaults to TRUE which means that packet buffering is enabled</td>
</tr>
<tr>
<td>MaximumBufferFileSize</td>
<td>Mandatory if BufferAccountingPackets is set to TRUE, determines the maximum buffer file size, defaults to 10 Megabyte)</td>
</tr>
<tr>
<td>NumberOfRetriesForBufferedPacket</td>
<td>Mandatory if BufferAccountingPackets is set to TRUE. A number greater than zero determines the number of attempts to be made to insert the buffered packet into Oracle. Defaults to 3.</td>
</tr>
<tr>
<td>BackingStoreEnvironmentVariables</td>
<td>Optional; when BufferAccountingPackets is set to TRUE, contains a comma-separated list of environment variable names to be stored into a local file along with buffered packet. No default. BackingStoreEnvironmentVariables can also be specified in scripts using the BackingStoreEnvironmentVariables environment variable.</td>
</tr>
<tr>
<td>UseLocalTimeZone</td>
<td>Set to TRUE or FALSE, determines the timezone of accounting records' TimeStamp (defaults to FALSE).</td>
</tr>
<tr>
<td>AttributeList</td>
<td>List of comma-separated attribute names.</td>
</tr>
<tr>
<td>Delimiter</td>
<td>Character used to separate the values of the attributes given in AttributeList property.</td>
</tr>
<tr>
<td>SQLDefinition</td>
<td>List of insert, update and delete statements to be executed to insert, update and delete the accounting record.</td>
</tr>
</tbody>
</table>

It is mandatory to set MaximumBufferFileSize property if BufferAccountingPackets property is set to TRUE. MaximumBufferFileSize can be specified in Kilobytes, Megabytes and Gigabytes. All values “512 kilobytes”, “512 k”, “512 KB” are valid for specifying 512 kilobytes.

If buffering is enabled, incoming packets will be accepted and logged to local file until the configured buffer file size is reached even if the database is offline. Attempts to insert them into Oracle will be made when database becomes available. This remote server will be marked as down only when the buffer gets
full. So, having two odbc-accounting remote servers in the service, first one with buffering enabled and multiple server policy of FailOver will make the other remote servers to receive packets only when the first remote server's buffer gets full.

AttributeList is to specify the list of attribute names separated with comma. When this 'AttributeList' is given in the MarkerList, these attributes' values will be appended together with delimiter specified in 'Delimiter' property and will be supplied as input to that marker.

Attributes from the Prime Access Registrar environment and request dictionaries can be specified in the MarkerList. Request dictionary will be looked up first for the attributes. Other than the standard attributes in the Prime Access Registrar dictionaries, two new marker variables are supported inside the marker list. They are,

- **TimeStamp**—Used to insert the timestamp into Oracle from Prime Access Registrar. Specifying this will supply the timestamp of that accounting record as a value to the insert statement. Time zone of this timestamp will be local if UseLocalTimeZone property is set to TRUE, otherwise GMT. This functionality could also be achieved by employing a trigger on the accounting table in the database. However, using this marker variable is recommended because the use of triggers negatively affects performance.

  The format of the timestamp marker variable supplied by Prime Access Registrar is **YYYYMMDDHH24MMSS**. For example, a timestamp of 20131107211050 represents 21:10:50, November 07, 2013.

- **RawAcctRecord**—Used to insert the entire accounting record into the database as a single text field. Contents of this will be whatever is sent by the NAS in the accounting packet and the format is **name=value** pairs delimited with the string specified in Delimiter property. If the delimiter property is not set, the default delimiter is a new line character. RawAcctRecord can be used with the other marker variables.

If multivalued attributes are specified in the marker list, the multiple values are concatenated together with delimiters, and the resulting value will be passed to the insert statement. This delimiter can be specified using the ODBCEnvironmentMultiValueDelimiter property under /Radius/Advanced.

### Configuration Examples

This section provides common Oracle accounting configuration examples most likely to be used.

This section contains the following topics:

- Inserting Selected Attributes into Separate Columns
- Inserting Complete Accounting Packets into One Column
- Inserting Selected Attributes into One Column
- Updating Selected Attributes
- Deleting Selected Attributes

#### Inserting Selected Attributes into Separate Columns

Use the following SQL and MarkerList properties statement to insert selected attributes into separate Oracle columns. The Oracle table definition will have separate columns for each attribute.

SQL: "insert into ar_acct (username, nasinfo, packet_type, timestamp) values (?, ?, ?, ?)"

MarkerList: "UserName/SQL_CHAR NAS-Identifier/SQL_CHAR Acct-Status-Type/SQL_CHAR TimeStamp/SQL_TIMESTAMP"
In this example, all the column data types are CHAR/VARCHAR except the timestamp which is DATE. If packet buffering option is disabled, instead of TimeStamp marker, you can also use Oracle's `sysdate` as a value for the timestamp column. The insert statement will look like the following:

```
"insert into ar_acct (username,nasinfo,packet_type,timestamp) values (?, ?, ?, sysdate)"
```

### Inserting Complete Accounting Packets into One Column

Use SQL and MarkerList properties in the SQLStatement like the following to insert the complete accounting packet into one Oracle column.

**SQL:** 
```
"insert into ar_acct (timestamp,raw_packet) values (?, ?)"
```

**MarkerList:** `"TimeStamp/SQL_TIMESTAMP RawAcctRecord/SQL_VARCHAR"`

### Inserting Selected Attributes into One Column

To insert selected attribute values into one Oracle column delimited by a comma (,), you must configure the AttributeList and Delimiter properties of the odbc-accounting RemoteServer object like the following:

**AttributeList** = "NAS-Identifier,NAS-Port,Acct-Status-Type,Acct-Session-Id"

**Delimiter** = ,

The SQL and MarkerList properties in the SQLStatement will look like the following:

**SQL:** 
```
"insert into ar_acct (username,timestamp,attributes) values (?, ?, ?)"
```

**MarkerList:** `"UserName/SQL_CHAR TimeStamp/SQL_TIMESTAMP AttributeList/SQL_VARCHAR"`

### Updating Selected Attributes

Use the following SQL and MarkerList properties statement to update the selected attributes:

**SQL:**  
```
"update arusers_acct set acct_status_type='stop' where username=? and acct_status_type=?"
```

**MarkerList:** `"UserName/SQL_CHAR Acct-Status-Type/SQL_CHAR"`

### Deleting Selected Attributes

Use the following SQL and MarkerList properties statement to delete the selected attributes:

**SQL:**  
```
"delete from arusers_acct where username=?"
```

**MarkerList:** `"UserName/SQL_CHAR"`

### Packet Buffering

You can optionally use packet buffering to improve latency when writing accounting records into the database. To enable packet buffering,

- set the BufferAccountingPackets property in the odbc-accounting remote server to TRUE.

This section contains the following topics:

- When Using Packet Buffering
- With Packet Buffering Disabled
When Using Packet Buffering

When BufferAccountingPackets is set to TRUE, the Prime Access Registrar server’s Accounting-Response is returned as soon as the accounting record is successfully written to the local file. To accomplish the queuing of accounting records to a local file, a variant of the existing session backing store is used.

- **Buffered packets** will be inserted into Oracle by a set of background worker threads. The Prime Access Registrar server tries to insert the buffered packet into Oracle for the number of retries configured in the NumberOfRetriesForBufferedPacket property (remote odbc accounting server definition). After the configured number of retries, the buffered packets are discarded from the local file.

- **Incoming packets** will be buffered to local file until the configured MaximumBufferFileSize is reached. After this limit is reached, no more packets will be addressed. When the database is offline, this remote server will continue to take incoming packets until MaximumBufferFileSize reaches. Prime Access Registrar tries to insert these buffered packets when database becomes available.

When using packet buffering, the Prime Access Registrar server can process more incoming packets and can reduce the bottleneck that could occur if the number of simultaneous incoming packets is large and the number of connections to the database is less.

With Packet Buffering Disabled

When BufferAccountingPackets is set to FALSE, Accounting-Response is returned after writing the accounting record into Oracle. Oracle write timing is immediate.

- Incoming packets are acknowledged by the remote server only after completing the write into Oracle.

- When the database is offline, no incoming packets are addressed. A slow database server impacts the packet processing rate.

Dynamic SQL Feature

Using this feature, you can choose the list of SQL statements and the sequence in which the SQL statements need to be executed during run time. This is done through the usage of scripting points.

The SQL-Sequence variable is provided in the Environment Dictionary and it takes the list of SQL statement names and separates each statement name by a semicolon (;). For example, the SQL statement names ‘sql3’, ‘sql4’, and ‘sql5’ are denoted as sql3;sql4;sql5;.

While being processed, the packet will be checked for the status of the SQL-Sequence variable. If the variable is set, the list of SQL statements will be executed in the order specified. Even if one of the SQL statements is not found in the configured list of SQL statements, the packet processing fails.

When configured for packet buffering, the BackingStore variable in the Environment Dictionary should have the SQL-Sequence variable in order to buffer the SQL-Sequence variable along with the packet information.
LDAP Accounting

Previous releases of Prime Access Registrar, supported accessing user data from an LDAP server, but this feature was limited to performing authentication and authorization (AA). You can only write the accounting records to local file or Oracle database or proxy to another RADIUS server.

Prime Access Registrar supports writing accounting records into LDAP server enabling integration between billing systems and LDAP.

- Prime Access Registrar adds a new type of service and remote server called ldap-accounting that enables inserting accounting records into LDAP.
- You can write accounting records into LDAP by referring this service in /Radius/DefaultAccountingService or in the Accounting-Service environment variable.

There is no specified schema structure to use the LDAP accounting feature. You can use your own object class design and configure, insert data using AttributesToWrite object in the Prime Access Registrar configuration. The Prime Access Registrar server inserts all configured attributes to write the accounting record into LDAP server. This feature is similar to the existing LDAP feature which performs authentication and authorization.

Note: Prime Access Registrar supports LDAP version 3 client and LDAP version 3 server.

Configuring LDAP Accounting

To use the ldap accounting feature,

- you must configure a service of type ldap-accounting under /Radius/Services.
- You must also configure at least one remote servers of type ldap-accounting under /Radius/RemoteServers.

This section contains the following topics:

- LDAP-Accounting Service
- LDAP RemoteServers
- Configuration Examples
- Configuring the LDAP Service for Accounting
- Configuring an LDAP-Accounting RemoteServer
- Setting LDAP-Accounting As Accounting Service

LDAP-Accounting Service

The following is an example of the LDAP-Accounting service:

```
[ //localhost/Radius/Services/ldap_accounting ]
Name = ldap_accounting
Description =
Type = ldap-accounting
IncomingScript~ =
OutgoingScript~ =
OutagePolicy~ = RejectAll
```
LDAP RemoteServers

Create a remote server under /Radius/RemoteServers, and set its protocol to ldap-accounting. The following is an example of an LDAP-Accounting RemoteServer’s configuration:

```
[ //localhost/Radius/RemoteServers/accounting_server ]
Name = accounting_server
Description =
Protocol = ldap-accounting
Port = 389
ReactivateTimerInterval = 300000
Timeout = 15
HostName =
BindName =
BindPassword =
UseSSL = FALSE
EnableKeepAlive = FALSE
DnPath =
EntryName = (uid=%s)
ObjectClass =
AttributeList =
Delimiter =
LDAPEnvironmentMultiValueDelimiter =
LimitOutstandingRequests = FALSE
MaxOutstandingRequests = 0
EscapeSpecialCharInUserName = FALSE
DNSLookupAndLDAPRebindInterval =
DataSourceConnections = 1
UseLocalTimeZone = FALSE
```

Table 7-2 lists the properties of LDAP-Accounting RemoteServer.

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the remote server; this property is mandatory and there is no default.</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of server.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Must be set to ldap-accounting.</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Mandatory time interval (in milliseconds) to activate an inactive server; defaults to 300000 ms.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Mandatory time interval (in seconds) to wait for LADP-write operation to complete; defaults to 15 seconds.</td>
</tr>
<tr>
<td>DataSourceConnections</td>
<td>Mandatory number of connections to be established; defaults to 8.</td>
</tr>
<tr>
<td>EnableKeepAlive</td>
<td>Required; default is FALSE. This is enabled to send a TCP keepalive to keep the idle connection active.</td>
</tr>
<tr>
<td>HostName</td>
<td>Required; the LDAP server’s hostname or IP address.</td>
</tr>
<tr>
<td><strong>Fields</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>BindName</td>
<td>Optional; the distinguished name (dn) to use when establishing a connection between the LDAP and RADIUS servers.</td>
</tr>
<tr>
<td>BindPassword</td>
<td>Optional; the password associated with the BindName.</td>
</tr>
<tr>
<td>DnPath</td>
<td>Required; the path that indicates where in the LDAP database to start the write for user information.</td>
</tr>
<tr>
<td>EntryName</td>
<td>Required; this specifies the write entry name. Prime Access Registrar uses when setting the LDAP server for user information. When you configure this property, use the notation &quot;%s&quot; to indicate where the user ID should be inserted. For example, a typical value for this property is &quot;(uid=%s)&quot;, which means that when setting for information about user joe, use the fentry name uid=joe.</td>
</tr>
<tr>
<td>UseLocalTimeZone</td>
<td>Optional; the default is FALSE. It determines the timezone of accounting records TimeStamp.</td>
</tr>
<tr>
<td>AttributeList</td>
<td>List of comma-separated attribute names.</td>
</tr>
<tr>
<td>Delimiter</td>
<td>Character used to separate the values of the attributes given in AttributeList property.</td>
</tr>
<tr>
<td>AttributesToWrite</td>
<td>List of inserts to be executed to insert the accounting record.</td>
</tr>
<tr>
<td>ObjectClass</td>
<td>Required; list of object classes which are all schemas defined in LDAP server. These schemas define required attributes and allowed attributes for an entry which is inserted from Prime Access Registrar.</td>
</tr>
<tr>
<td>LDAPEnvironmentMultiValueDelimiter</td>
<td>Optional; allows you to specify a character that separates multi-valued attribute lists when using ldap-accounting.</td>
</tr>
<tr>
<td>LimitOutstandingRequests</td>
<td>Required; the default is FALSE. Prime Access Registrar uses this property in conjunction with the MaxOutstandingRequests property to tune the RADIUS server's use of the LDAP server. When you set this property to TRUE, the number of outstanding requests for this RemoteServer is limited to the value you specified in MaxOutstandingRequests. When the number of requests exceeds this number, Prime Access Registrar queues the remaining requests, and sends them as soon as the number of outstanding requests drops to this number.</td>
</tr>
<tr>
<td>MaxOutstandingRequests</td>
<td>Required when you have set the LimitOutstandingRequests to TRUE. The number you specify, which must be greater than zero, determines the maximum number of outstanding requests allowed for this remote server.</td>
</tr>
<tr>
<td>EscapeSpecialCharInUserName</td>
<td>FALSE by default.</td>
</tr>
<tr>
<td>UseSSL</td>
<td>A boolean field indicating whether you want Prime Access Registrar to use SSL (Secure Socket Layer) when communicating with this RemoteServer. When you set it to TRUE, be sure to specify the CertificateDBPath field in the Advanced section, and be sure the port you specified for this RemoteServer is the SSL port used by the LDAP server.</td>
</tr>
</tbody>
</table>
AttributeList is to specify the list of attribute names separated with comma. When this 'AttributeList' is given in the 'AttributesToWrite' object, these attribute values will be appended together with delimiter specified in 'Delimiter' property and will be supplied as input to that ldap field name.

Attributes from the Prime Access Registrar environment and request dictionaries can be specified in the 'AttributesToWrite' object. Request dictionary will be looked up first for the attributes. Other than the standard attributes in the Prime Access Registrar dictionaries, two new variables are supported inside the 'AttributesToWrite' object.

They are:

- **TimeStamp**—Used to insert the timestamp into LDAP server from Prime Access Registrar.
  Specifying this will supply the timestamp of that accounting record as a value to the insert. Time zone of this timestamp will be local if UseLocalTimeZone property is set to TRUE, otherwise GMT. This functionality could also be achieved by employing a trigger on the accounting object class in the server.

  The format of the timestamp variable supplied by Prime Access Registrar is **YYYYMMDDHH24MMSS**. For example, a timestamp of 20131107211050 represents 21:10:50, November 07, 2013.

- **RawAcctRecord**—Used to insert the entire accounting record into the database as a single text field. Contents of this will be whatever is sent by the NAS in the accounting packet and the format is name=value pairs delimited with the string specified in Delimiter property. If the delimiter property is not set, the default delimiter is a ‘,’ character. RawAcctRecord can be used with the other variables.

  If multivalued attributes are specified in the attribute list, the multiple values are concatenated together with delimiters, and the resulting value will be passed to the insert statement. This delimiter can be specified using the LDAPEnvironmentMultiValueDelimiter property.

### Configuration Examples

This section provides common LDAP accounting configuration examples most likely to be used.

This section contains the following topics:

- Inserting Selected Attributes into Separate LDAP Field
- Inserting Complete Accounting Packets into One Field
- Inserting Selected Attributes into One Field

### Inserting Selected Attributes into Separate LDAP Field

Use the following ObjectClass property and ‘AttributesToWrite’ object properties statement to insert selected attributes into separate LDAP schema. The LDAP schema definition will have separate fields for each attribute.

```
[ //localhost/Radius/RemoteServers/accounting-server/AttributesToWrite ]
  sn = timestamp
  uid = username
```

### Inserting Complete Accounting Packets into One Field

Use ObjectClass and ‘AttributesToWrite’ object properties in the ldap-accounting remote server like the following to insert the complete accounting packet into one LDAP field.

```
[ //localhost/Radius/RemoteServers/accounting-server/AttributeWrites ]
  seealso = rawacctrecord
```
uid = username

Inserting Selected Attributes into One Field

To insert selected attribute values into one LDAP field delimited by a comma (,), you must configure the AttributeList and Delimiter properties of the ldap-accounting RemoteServer object like the following:

AttributeList = User-Name,NAS-Port,Acct-Session-Id
Delimiter = ,
AttributeWrites/
telephonenumber = attributelist
uid = username

Configuring the LDAP Service for Accounting

You configure an LDAP-Accounting service under /Radius/Services. When you define an LDAP-Accounting service under /Radius/Services, you must set its type to ldap-accounting.

```
[ //localhost/Radius/Services/AR-LDAP-ACCT ]
  Name = AR-LDAP-ACCT
  Description =
  Type = ldap-accounting
  IncomingScript- =
  OutgoingScript- =
  OutagePolicy- = RejectAll
  OutageScript- =
  MultipleServersPolicy = Failover
  Remoteservers/
```

Table 7-3  LDAP-Accounting Service Properties

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; inherited from the upper directory.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the service.</td>
</tr>
<tr>
<td>Type</td>
<td>Must be set to LDAP for LDAP service.</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional.</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; must be set to AcceptAll, DropPacket, or RejectAll. Default is DropPacket.</td>
</tr>
<tr>
<td>OutageScript</td>
<td>Optional. if you set this property to the name of a script, Prime Access Registrar runs it when an outage occurs. This property allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
<tr>
<td>MultipleServersPolicy</td>
<td>Required; must be set to RoundRobin or defaults to Failover.</td>
</tr>
<tr>
<td>RemoteServers</td>
<td>Required; list of one or more remote servers defined under /Radius/Services/LDAP/RemoteServers. These servers must be listed in.</td>
</tr>
</tbody>
</table>

This section contains the following topics:

- MultipleServersPolicy
- RemoteServers
MultipleServersPolicy

Use the MultipleServersPolicy property to configure the LDAP remote servers in RoundRobin mode, or the default Failover mode applies. When set to Failover, Prime Access Registrar directs requests to the first server in the /Radius/Services/LDAP/RemoteServers list. If that server should fail or go offline, Prime Access Registrar redirects all requests to the next server in the list. The process continues until Prime Access Registrar locates an online server.

When set to RoundRobin, Prime Access Registrar directs each request to the next server in the RemoteServers list to share the resource load across all listed servers.

RemoteServers

Use the RemoteServers directory to list one or more remote servers to process access requests. The servers must also be listed in order under /Radius/RemoteServers.

The order of the RemoteServers list determines the sequence for directing access requests when MultipleServersPolicy is set to RoundRobin mode. The first server in the list receives all access requests when MultipleServersPolicy is set to Failover mode.

Configuring an LDAP-Accounting RemoteServer


The Name, Protocol, Port, HostName, BindName, BindPassword, DnPath, and EntryName properties must be configured to use an LDAP remote server.

<table>
<thead>
<tr>
<th>Table 7-4 LDAP Remote Server Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Protocol</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
</tr>
<tr>
<td>Timeout</td>
</tr>
<tr>
<td>DataSourceConnections</td>
</tr>
<tr>
<td>EnableKeepAlive</td>
</tr>
<tr>
<td>HostName</td>
</tr>
<tr>
<td>BindName</td>
</tr>
<tr>
<td>BindPassword</td>
</tr>
<tr>
<td>DnPath</td>
</tr>
</tbody>
</table>
LDAP Accounting

Chapter 7  RADIUS Accounting

LDAP Accounting

Table 7-4  LDAP Remote Server Properties (continued)

<table>
<thead>
<tr>
<th>Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EntryName</td>
<td>Required; this specifies the write entry name Prime Access Registrar uses when inserting the LDAP server for user information. When you configure this property, use the notation &quot;%s&quot; to indicate where the user ID should be inserted. For example, a typical value for this property is &quot;(uid=%s),&quot; which means that when inserting for information about user joe, use the entry name uid=joe.</td>
</tr>
<tr>
<td>UseLocalTimeZone</td>
<td>Set to TRUE or FALSE, determines the timezone of accounting records’ TimeStamp (defaults to FALSE).</td>
</tr>
<tr>
<td>AttributeList</td>
<td>List of comma-separated attribute names.</td>
</tr>
<tr>
<td>Delimiter</td>
<td>Character used to separate the values of the attributes given in AttributeList property.</td>
</tr>
<tr>
<td>AttributesToWrite</td>
<td>List of inserts to be executed to insert the accounting record.</td>
</tr>
<tr>
<td>ObjectClass</td>
<td>Required; list of object classes which are all schemas defined in LDAP server. These schemas define required attributes and allowed attributes for an entry which is inserted from Prime Access Registrar.</td>
</tr>
<tr>
<td>LDAPEnvironmentMultiValueDelimiter</td>
<td>Optional; allows you to specify a character that separates multi-valued attribute lists when using ldap-accounting.</td>
</tr>
<tr>
<td>LimitOutstandingRequests</td>
<td>Required; the default is FALSE. Prime Access Registrar uses this property in conjunction with the MaxOutstandingRequests property to tune the RADIUS server’s use of the LDAP server. When you set this property to TRUE, the number of outstanding requests for this RemoteServer is limited to the value you specified in MaxOutstandingRequests. When the number of requests exceeds this number, Prime Access Registrar queues the remaining requests, and sends them as soon as the number of outstanding requests drops to this number.</td>
</tr>
<tr>
<td>MaxOutstandingRequests</td>
<td>Required when you have set the LimitOutstandingRequests to TRUE. The number you specify, which must be greater than zero, determines the maximum number of outstanding requests allowed for this remote server.</td>
</tr>
<tr>
<td>EscapeSpecialCharInUserName</td>
<td>FALSE by default.</td>
</tr>
<tr>
<td>UseSSL</td>
<td>A boolean field indicating whether you want Prime Access Registrar to use SSL (Secure Socket Layer) when communicating with this RemoteServer. When you set it to TRUE, be sure to specify the CertificateDBPath field in the Advanced section, and be sure the port you specified for this RemoteServer is the SSL port used by the LDAP server.</td>
</tr>
</tbody>
</table>

DNS Look Up and LDAP Rebind Interval

Prime Access Registrar provides a DNS Look-up and LDAP Rebind feature that enables you to use a smart DNS server for LDAP hostname resolution, allows you to query a DNS server at set intervals to resolve the LDAP hostname, and optionally rebind to the LDAP server, if necessary.
When you configure Prime Access Registrar to use an LDAP directory server, you can specify the hostname of the LDAP directory server. The hostname can be a qualified or an unqualified name. You can also specify a timeout period after which Prime Access Registrar will again resolve the hostname. If the IP address returned is different from the previous, Prime Access Registrar establishes a new LDAP bind connection.

The DNSLookupAndLDAPRebindInterval property specifies the timeout period after which the Prime Access Registrar server will attempt to resolve the LDAP hostname to IP address (DNS resolution). When you do not modify DNSLookupAndLDAPRebindInterval, the default value zero indicates the server will perform normal connection and binding only at start-up time or during a reload. Unless you change the default to a value greater than zero, the server will not perform periodic DNS lookups.

Prime Access Registrar maintains and uses the existing bind connection until a new one is established to minimize any performance impact during the transfer. Prime Access Registrar ensures that no requests are dropped or lost during the transfer to a new LDAP binding.

Set the DNSLookupAndLDAPRebindInterval using a numerical value and the letter H for hours or M for minutes, such as in the following examples:

```
set DNSLookupAndLDAPRebindInterval 15M — performs DNS resolution every 15 minutes
```

**Note**

We recommend that you do not set DNSLookupAndLDAPRebindInterval to a value less than 15 minutes to minimize its effect on server performance.

```
set DNSLookupAndLDAPRebindInterval 1h — performs DNS resolution every hour
```

**Configuring the DNS Look-up and LDAP Rebind**

To configure the DNS Look-up and LDAP Rebind:

**Step 1**

Log into the Prime Access Registrar server, and use `aregcmd` to navigate to `//localhost/Radius/Remoteservers`. If necessary, add the LDAP server, or change directory to it.

```
  cd /Radius/RemoteServers/ldap-serv1/
```

**Step 2**

Set the DNSLookupAndLDAPRebindInterval property to the interval time desired.

```
  set DNSLookupAndLDAPRebindInterval 30M
```

**LDAP Rebind Failures**

Prime Access Registrar records any name resolution failures, bind successes and failures, and the destination hostname and IP address in the log file. At trace level 3, Prime Access Registrar also logs the time of any new bind connections and the closing of any old bind connections.

If either the name resolution or bind attempt fail, Prime Access Registrar continues using the existing bind connection until the timeout has expired again. If there is no existing bind connection, Prime Access Registrar marks the remote server object as `down`.

**Setting LDAP-Accounting As Accounting Service**

Use `aregcmd` to configure the LDAP-accounting Service as the default accounting service under `/Radius` as in the following:
MySQL Support

Prime Access Registrar provides support for MySQL to query user records from a MySQL database using odbc interface and enables you to write accounting records into MySQL database using odbc-accounting. Prime Access Registrar has been tested with MySQL 5.0.90 and MyODBC 3.51.27 (reentrant).

For the Prime Access Registrar server to use MySQL, you must create and configure an ODBCDataSource object of type myodbc and a RemoteServer object set to protocol odbc.

**Note**

For more information about dynamic SQL feature, see Dynamic SQL Feature, page 7-11.

This section contains the following topics:

- Configuring MySQL
- Example Configuration

Configuring MySQL

To configure the Prime Access Registrar server to query records form a MySQL database:

**Step 1** Log into the Prime Access Registrar server and launch `aregcmd`.
Log in as a user with administrative rights such as user `admin`.

**Step 2** Change directory to the `/Radius/Advanced/ODBCDataSources` and add a new ODBCDataSource.

```bash
cd /Radius/Advanced/ODBCDataSources
add mysql
```

**Step 3** Set the new ODBCDatasource type to myodbc.

```bash
cd mysql
set type myodbc
```

**Step 4** Set the Driver property to the path of the MyODBC library.

**Step 5** Set the UserID property to a valid username for the MyODBC database and provide a valid password for this user.

**Step 6** Provide a DataBase name and the name of the Prime Access Registrar RemoteServer object to associate with the ODBCDataSource.

**Step 7** Change directory to `/Radius/RemoteServers` and add a RemoteServer object to associate with the new ODBCDataSource.

```bash
cd /Radius/RemoteServers
add mysql
```
Step 8  Change directory to the new RemoteServer and set its protocol to odbc-accounting.

```
cd mysql
set protocol odbc-accounting
```

Step 9  Set the ODBCDataSource property to the name of the ODBCDataSource to associate with this RemoteServer object.

```
set ODBCDataSource mysql
```

Example Configuration

The following shows an example configuration for a MySQL ODBC data source.

```
[ //localhost/Radius/Advanced/ODBCDataSources/mysql ]
  Name = mysql
  Type = myodbc
  Driver = /tmp/libmyodbc3_r.so
  UserID = mysql
  Password = <encrypted>
  DataBase = test
  Server = mysql-a
  Port = 3306
```

The following shows an example configuration for a RemoteServer

```
  Name = odbc-accounting
  Description =
  Protocol = odbc-accounting
  ReactivateTimerInterval = 300000
  Timeout = 15
  DataSourceConnections = 8
  ODBCDataSource =
  KeepAliveTimerInterval = 0
  BufferAccountingPackets = TRUE
  MaximumBufferSize = "10 Megabytes"
  NumberOfRetriesForBufferedPacket = 3
  BackingStoreEnvironmentVariables =
  UseLocalTimeZone = FALSE
  AttributeList =
  Delimiter =
  SQLDefinition/
  ODBCToRadiusMappings/
  ODBCToEnvironmentMappings/
  ODBCToCheckItemMappings/
```

Proxying Accounting Records

You can configure Prime Access Registrar to store accounting records locally and to proxy the accounting records to a remote RADIUS server thereby maintaining multiple accounting logs.

This section contains the following topics:

- Configuring the Local Cisco Prime Access Registrar Server
Configuring the RemoteServer Object

Configuring the Local Cisco Prime Access Registrar Server

This type of setup requires you to configure the following on the local Prime Access Registrar server:

- A local accounting service of type file
- A remote accounting service of type radius
- An accounting service of type group
- A RemoteServer object

This section contains the following topics:

- Configuring the Local Accounting Service
- Configuring the Remote Accounting Service
- Configuring the Group Accounting Service

Configuring the Local Accounting Service

The following example shows the configuration required for a local accounting service. This service must be of type file.

```
[//localhost/Radius/Services/accserv1/ ]
Name = accserv1
Description =
Type = file
IncomingScript~ =
OutgoingScript~ =
OutagePolicy~ = RejectAll
OutageScript~ =
FilenamePrefix = accounting
MaxFileSize = "10 Megabytes"
MaxFileAge = "1 Day"
RolloverSchedule =
UseLocalTimeZone = FALSE
```

Configuring the Remote Accounting Service

The following example shows the configuration required for a remote accounting service. This service must be of type `radius`, and the name of the remote server must be listed under the RemoteServers subdirectory.

```
[//localhost/Radius/Services/accserv2/]
Name = accserv2
Description =
Type = radius
IncomingScript~ =
OutgoingScript~ =
OutagePolicy~ = RejectAll
OutageScript~ =
FilenamePrefix = accounting
MaxFileSize = "10 Megabytes"
MaxFileAge = "1 Day"
RemoteServers/
    1. RemoteRADIUS
```
Configuring the Group Accounting Service

The following example shows the configuration required for a grouping accounting service. This service must be of type group and the local and remote accounting services, accserv1 and accserv2 in the previous examples, should be added under the GroupServices subdirectory.

The CiscoAccounting service groups these two services. The type property should be set to group. The services accserv1 and accserv2 should be added under GroupServices subdirectory of CiscoAccounting service.

```plaintext
[//localhost/Radius/Services/GroupAccounting/
   Name = GroupAccounting
   Description =
   Type = group
   IncomingScript~ =
   OutgoingScript~ =
   RolloverSchedule =
   ResultRule = AND
   GroupServices/
   1. accserv1
   2. accserv2
```

Refer to Service Grouping Feature, page 17-14, for more information about the Prime Access Registrar Service Grouping feature.

Configuring the RemoteServer Object

The following example shows the configuration required for the RemoteServer object in the local Prime Access Registrar server.

```plaintext
[ //localhost/Radius/RemoteServers ]
   Entries 1 to 1 from 1 total entries
   Current filter: <all>

RemoteRADIUS/
   Name = RemoteRADIUS
   Description =
   Protocol = radius
   IPAddress = aa.bb.cc.dd
   Port = 1812
   ReactivateTimerInterval = 300000
   SharedSecret = secret
   Vendor =
   IncomingScript~ =
   OutgoingScript~ =
   MaxTries = 3
   InitialTimeout = 2000
   AccountingPort = 1813
   ACKAccounting = TRUE
```

If the ACKAccounting property is set to FALSE, Prime Access Registrar disregards the accounting acknowledgment and continues with the packet processing rather than waiting for the accounting acknowledgment from the Remote server.

If the ACKAccounting property is set to FALSE, Prime Access Registrar provides the SendandForget option. You can set this option to TRUE, to delete the original and proxy requests from the buffer that Prime Access Registrar maintains after sending an accounting response to the client.
The group service, CiscoAccounting in this example, should be defined as the default accounting service for any accounting packets received by the local Prime Access Registrar server, as in the following:

```
set /Radius/DefaultAccountingService CiscoAccounting
```

**Accounting Log Examples**

This section provides examples of accounting log information recorded in an accounting log file. This section contains the following topics:

- Accounting-Start Packet
- Accounting Stop Packet
- Trace of Successful Accounting

**Accounting-Start Packet**

The Accounting-Start packet describes the type of service and the user attempting to login.

```
Tue, 06 Dec 2013 12:32:17.036
User-Name = bob
NAS-Port = 1
Framed-IP-Address = 1.1.1.1
Class = yahoo.com
NAS-Identifier = localhost
Acct-Status-Type = Start
Acct-Session-Id = 1
```

**Accounting Stop Packet**

When the session ends, the NAS sends an Accounting Stop packet that describes the type of service that was delivered. The Accounting Stop packet might also contain statistics such as elapsed time, input and output octets, or input and output packets.

```
Tue, 06 Dec 2013 12:32:17.036
User-Name = bob
NAS-Port = 1
Framed-IP-Address = 1.1.1.1
Class = yahoo.com
NAS-Identifier = localhost
Acct-Status-Type = Stop
Acct-Session-Id = S209524
```

**Trace of Successful Accounting**

The following is a trace example of a successful accounting sequence.

```
11/12/2013 21:27:58: P6699: Packet received from 10.1.9.204
```
Sample Error Messages

The following are sample accounting error messages:

Error message logged in name_radius_1_log file when the disk is full and AR is trying to record an accounting request.

05/15/2013 2:52:29 name/radius/1 Error System 0 Failed to write records to the accounting report file '/usr/accounting.log' - accounting records lost

Note: An Accounting-Response packet is sent only if the accounting record is written to the file in the disk. If the disk is full, an Accounting-Response packet is not sent.

Error message logged in name_radius_1_log file when the path specified in the FilenamePrefix property is not valid.

05/15/2013 4:11:12 name/radius/1 Error Configuration 0 Error in property /Radius/Services/CiscoAccounting/FilenamePrefix: Unable to write to the specified report file prefix (/tmp/AR/accounting)
Diameter

Diameter is a networking protocol which is derived from RADIUS protocol. It is considered to be the next generation Authentication, Authorization, and Accounting (AAA) protocol. This is the other core protocol used in the IP Multimedia Subsystem (IMS) architecture for IMS Entities to exchange AAA related information. Cisco Prime Access Registrar (Prime Access Registrar) supports Diameter Applications based on the Diameter Base Protocol defined in RFC 6733.

Diameter is composed of a base protocol and a set of applications which allows it to extend its services to new access technologies. The base protocol provides basic mechanisms for reliable transport, message delivery, and error handling. Each application is defined by an application identifier and associated with commands. Each command is defined with mandatory Attribute Value Pairs (AVPs) and non-mandatory AVPs including vendor-specific AVPs.

The base protocol must be used in conjunction with a Diameter application. Each application relies on the services of the base protocol to support a specific type of network access.

The following is the list of applications supported by Prime Access Registrar:

- Diameter Network Access Server Application (NASREQ, RFC 4005)
- Diameter Base Accounting (RFC 6733)
- Diameter Extensible Authentication Protocol (EAP) Application (RFC 4072)

This chapter contains the following sections:

- Diameter with EAP Support, page 8-2
- Diameter Server Startup Log, page 8-3
- Diameter Stack Level Messages, page 8-4
- Configuring Authentication and Authorization for Diameter, page 8-6
- Configuring the Diameter Application in Prime Access Registrar, page 8-8
- Writing Diameter Application in Prime Access Registrar, page 8-15
- Translation Framework for Diameter, page 8-19
- TLS Support for Diameter, page 8-20
- Managing Diameter Sessions, page 8-22
Diameter with EAP Support

The Extensible Authentication Protocol (EAP), is an authentication framework which supports multiple authentication mechanisms. EAP may be used on dedicated links, switched circuits, and wired as well as wireless links. For more information on EAP support in Prime Access Registrar, see Chapter 9, “Extensible Authentication Protocols.”

Prime Access Registrar supports Diameter EAP application that carries EAP packets between a Network Access Server (NAS) working as an EAP Authenticator and a back-end authentication server. The Diameter EAP application is based on the Diameter Network Access Server Application [NASREQ] and is intended for environments similar to NASREQ.

In the Diameter EAP application, authentication occurs between the EAP client and its home Diameter server. This end-to-end authentication reduces the possibility for fraudulent authentication, such as replay and man-in-the-middle attacks. End-to-end authentication also provides a possibility for mutual authentication, which is not possible with PAP and CHAP in a roaming PPP environment.

This topic contains the following sections:

- Advertising Application Support, page 8-2
- Diameter EAP Conversation Flow, page 8-2

Advertising Application Support

Diameter nodes conforming to this specification must advertise support by including the Diameter EAP Application ID value of 5 in the Auth-Application-Id AVP of the Capabilities-Exchange-Request and Capabilities-Exchange-Answer command [BASE].

If the NAS receives a response with the Result-Code set to DIAMETER_APPLICATION_UNSUPPORTED [BASE], it indicates that the Diameter server in the home realm does not support EAP. If possible, the access device may attempt to negotiate another authentication protocol, such as PAP or CHAP. An access device must be cautious when determining whether a less secure authentication protocol will be used, since this could result from a downgrade attack.

Diameter EAP Conversation Flow

The EAP conversation between the authenticating peer and the access device begins with the initiation of EAP within a link layer, such as PPP [RFC1661] or IEEE 802.11i [IEEE-802.11i]]. Once EAP has been initiated, the access device will typically send a Diameter-EAP-Request message with an empty EAP-Payload AVP to the Diameter server, signifying an EAP-Start. Prime Access Registrar routes the message to the Diameter EAP service through the rules and policy engine (and/or client, server and vendor scripting point) through which the packet is processed. The Diameter EAP Service forms a Diameter-EAP-Answer message containing an EAP-Payload AVP that includes an encapsulated EAP packet. The Result-Code AVP in the message will be set to DIAMETER_MULTI_ROUND_AUTH, signifying that a subsequent request is expected.

Figure 8-1 describes the Diameter EAP request flow.
The access device issues the EAP-Request/Identity message to the EAP client, and forwards the EAP-Response/Identity packet, encapsulated within the EAP-Payload AVP, as a Diameter-EAP-Request to Prime Access Registrar as shown in Figure 8-2. This reduces the number of Diameter message round trips.

The conversation continues until the Diameter server sends a Diameter-EAP-Answer with a Result-Code AVP indicating success or failure, and an optional EAP-Payload. The Result-Code AVP is used by the access device to determine whether service is to be provided to the EAP client or not. The access device must not rely on the contents of the optional EAP-Payload to determine whether service is to be provided or not.

**Diameter Server Startup Log**

When Prime Access Registrar starts, Diameter server also starts.

The log file shows the following:

```
09/30/2013 6:38:47.419 name/radius/1 Info Server 0 Diameter Server Started
09/30/2013 6:38:47.437 name/radius/1 Info Protocol 0 Starting diameter core
```
The following are the stack level messages that are exchanged between the diameter peers:

- **Capabilities Exchange Message**
- **Watchdog Message**
- **Disconnect Message**
Capabilities Exchange Message

When Diameter peers establish a transport connection to Prime Access Registrar, they will exchange the Capabilities Exchange messages. This message allows the discovery of a peer's identity and its capabilities (protocol version number, supported Diameter applications, security mechanisms, etc.)

The log file shows the following:

```plaintext
05/14/2015  5:36:19.646 name/radius/1 Info Server 0 Starting Server
05/14/2015  5:36:19.707 name/radius/1 Info Server 0 RollingEncryption using new key 17 and aging key 18
05/14/2015  5:36:19.732 name/radius/1 Info Server 0 RollingEncryption using new key 17 and aging key 18
05/14/2015  5:36:19.852 name/radius/1 Info Server 0 Device-Watchdog-Request thread activated for peer 10.81.79.60
05/14/2015  5:36:19.852 name/radius/1 Info System 0 Connecting to 10.81.79.60:4000...
05/14/2015  5:36:19.852 name/radius/1 Info System 0 Local socket bind on 10.81.79.81:0
05/14/2015  5:36:20.852 name/radius/1 Info Protocol 0 Connected to RemoteServer b2
05/14/2015  5:36:20.853 name/radius/1 Info Server 0 Initiating CER to 10.81.79.60...
05/14/2015  5:36:20.853 name/radius/1 Info Server 0 Received CEA from the peer(10.81.79.60), IP: 10.77.240.41
05/14/2015  5:36:20.854 name/radius/1 Info Server 0 Capabilities are successfully exchanged with 10.81.79.60
05/14/2015  5:36:21.053 name/radius/1 Info Server 0 Sticky Sessions BGwrite thread activated
05/14/2015  5:36:21.053 name/radius/1 Info Server 0 Sticky Session Count BG thread activated.
05/14/2015  5:36:21.058 name/radius/1 Info Server 0 Starting Interface 127.0.0.1, port 1812 (RADIUS Access)
05/14/2015  5:36:21.058 name/radius/1 Info Server 0 Starting Interface 127.0.0.1, port 1813 (RADIUS Accounting)
05/14/2015  5:36:21.058 name/radius/1 Info Server 0 Starting Interface 127.0.0.1, port 49 (TACACS+)
05/14/2015  5:36:21.059 name/radius/1 Info Server 0 Starting Interface 127.0.0.1, port 3868 (Diameter-TCP)
05/14/2015  5:36:21.059 name/radius/1 Info Server 0 Starting Interface 127.0.0.1, port 3868 (Diameter-SCTP)
05/14/2015  5:36:21.059 name/radius/1 Info Server 0 Starting Interface 10.81.79.81, port 1812 (RADIUS Access)
05/14/2015  5:36:21.059 name/radius/1 Info Server 0 Starting Interface 10.81.79.81, port 1813 (RADIUS Accounting)
05/14/2015  5:36:21.059 name/radius/1 Info Server 0 Starting Interface 10.81.79.81, port 49 (TACACS+)
05/14/2015  5:36:21.059 name/radius/1 Info Server 0 Starting Interface 10.81.79.81, port 3868 (Diameter-TCP)
05/14/2015  5:36:21.059 name/radius/1 Info Server 0 Starting Interface 10.81.79.81, port 3868 (Diameter-SCTP)
05/14/2015  5:36:21.060 name/radius/1 Error Configuration 0 Interface fe80::56ff:fe99:9e20: af_bind() to port 1812 failed with -2147418090
05/14/2015  5:36:21.060 name/radius/1 Error Server 0 Failed to start IPv6 Interface fe80::56ff:fe99:9e20, port 1812 (RADIUS Access)
05/14/2015  5:36:21.060 name/radius/1 Info Server 0 Starting Replication Manager
```
Watchdog Message

The Device-Watchdog-Request and Device-Watchdog-Answer messages are used to proactively detect transport failures. Device Watchdog message time interval is configurable in Prime Access Registrar.

Disconnect Message

Disconnect messages are initiated when Diameter peers lose transport connection to Prime Access Registrar.

Configuring Authentication and Authorization for Diameter

This section describes how to configure Prime Access Registrar to perform authentication and authorization and how to configure a local service and userlist.

See Table 4-7 for more information on Diameter client properties.

This section contains the following topics:

- Configuring Local Authentication and Authorization
- Configuring External Authentication Service

Configuring Local Authentication and Authorization

In Diameter, an AA-Request packet is a request for authentication and authorization. Authentication checks username and password credentials, while authorization typically involves returning the correct information to allow the service a user is authorized to have. Prime Access Registrar performs AA and returns the appropriate Diameter attributes in an AA-Answer packet.

For adding a Diameter peer in Prime Access Registrar, configure a new entry in the clients (including Policy and Charging Rules Functions (PCRF), Home Subscriber Servers (HSS), Mobility Management Entities (MME), Online Charging Systems (OCS), and others) and remote server object.

The following shows an example configuration for adding a Diameter peer (NAS/Client) in Prime Access Registrar.

```
[ //localhost/Radius/Clients/70dia ]
Name = 70dia
Description =
Protocol = diameter
HostName = 10.81.79.241
PeerPort = 3868
Vendor =
IncomingScript~ =
OutgoingScript~ =
AdvertisedHostName =
```
AdvertisedRealm =
InitialTimeout = 1000
MaxIncomingRequestRate = 0
WatchDogTimeout = 500
SCTP-Enabled = FALSE
TLS-Enabled = FALSE

[ //localhost/Radius/Services/diaservice ]
Name = diaservice
Description =
Type = diameter
IncomingScript~ =
OutgoingScript~ =
EnableSticky = FALSE
MultiplePeersPolicy = Failover
PeerTimeOutPolicy = FailOver
DiaRemoteServers/
Entries 1 to 1 from 1 total entries
Current filter: <all>

65/
Name = 65
Metric = 2
Weight = 0
IsActive = TRUE

Note
You should restart the Prime Access Registrar server if you change any Diameter specific configuration.

See Table 4-7 and Table 4-21 for more details.

Configuring a Local Service and UserList

See “Configuring a Local Service and UserList” section on page 6-1 for more information on how to configure a local service and user list.

The following messages are logged in the trace file at the time of authenticating a valid user:

05/14/2015  5:41:47.574: P734: Packet received from 10.81.79.81
05/14/2015  5:41:47.574: P734: Application id: 1, Cmd code: 265, Flag: 0x80
05/14/2015  5:41:47.574: P734: Using Client: vm050
05/14/2015  5:41:47.574: P734: Packet successfully added
05/14/2015  5:41:47.574: P734: Trace of Diameter Packet
05/14/2015  5:41:47.574: P734: Destination-Realm = cisco.com
05/14/2015  5:41:47.574: P734: User-Name = bob
05/14/2015  5:41:47.574: P734: User-Password = <encrypted>
05/14/2015  5:41:47.574: P734: Auth-Request-Type = AUTHORIZE_ONLY
05/14/2015  5:41:47.574: P734: Origin-Host = ar-lnx-vm050.cisco.com
05/14/2015  5:41:47.574: P734: Session-Id = .2096298391;2
05/14/2015  5:41:47.574: P734: Auth-Application-Id = 1
05/14/2015  5:41:47.574: P734: Origin-Realm = xyz.com
05/14/2015  5:41:47.574: P734: Tracing the packet after running the rules and policies
05/14/2015  5:41:47.574: P734: Using Client: vm050
05/14/2015  5:41:47.574: P734: FastRule Engine called for packet
05/14/2015  5:41:47.574: P734: Fastrule return = 0
05/14/2015  5:41:47.574: P734: Authorizing with Service local-users
05/14/2015  5:41:47.574: P734: Getting User bob’s UserRecord from UserList Default
05/14/2015  5:41:47.574: P734: User bob is part of UserGroup PPP-users
05/14/2015  5:41:47.574: P734: Merging UserGroup PPP-users’s BaseProfiles into response dictionary
05/14/2015 5:41:47.575: P734: Merging UserGroup PPP-users’s Attributes into response Dictionary
05/14/2015 5:41:47.575: P734: Merging attributes into the Response Dictionary:
05/14/2015 5:41:47.575: P734: Trace of Diameter Packet
05/14/2015 5:41:47.575: P734: User-Name = bob
05/14/2015 5:41:47.575: P734: Result-Code = Diameter-Success
05/14/2015 5:41:47.575: P734: Auth-Request-Type = AUTHORIZE_ONLY
05/14/2015 5:41:47.575: P734: Origin-Host = 10.81.79.81
05/14/2015 5:41:47.575: P734: Session-Id = .;2096298391;2
05/14/2015 5:41:47.575: P734: Auth-Application-Id = 1
05/14/2015 5:41:47.575: P734: Origin-Realm = cisco.com
05/14/2015 5:41:47.575: P734: Sending response to ar-lnx-vm050.cisco.com
05/14/2015 5:41:47.575: P734: Packet successfully removed
05/14/2015 5:41:47.575: P734: Packet Deleted

The following messages are logged in the trace file at the time of authenticating an invalid user:

05/14/2015 5:45:29.478: P831: Packet received from 10.81.79.81
05/14/2015 5:45:29.478: P831: Application id: 1, Cmd code: 265, Flag: 0x80
05/14/2015 5:45:29.478: P831: Using Client: vm050
05/14/2015 5:45:29.478: P831: Packet successfully added
05/14/2015 5:45:29.478: P831: Trace of Diameter Packet
05/14/2015 5:45:29.478: P831: Destination-Realm = cisco.com
05/14/2015 5:45:29.478: P831: User-Name = user.1
05/14/2015 5:45:29.478: P831: User-Password = <encrypted>
05/14/2015 5:45:29.478: P831: Auth-Request-Type = AUTHORIZATION
05/14/2015 5:45:29.479: P831: Origin-Host = ar-lnx-vm050.cisco.com
05/14/2015 5:45:29.479: P831: Session-Id = .;2096298391;3
05/14/2015 5:45:29.479: P831: Auth-Application-Id = 1
05/14/2015 5:45:29.479: P831: Origin-Realm = xyz.com
05/14/2015 5:45:29.479: P831: Tracing the packet after running the rules and policies
05/14/2015 5:45:29.479: P831: Using Client: vm050
05/14/2015 5:45:29.479: P831: FastRule Engine called for packet
05/14/2015 5:45:29.479: P831: Fastrule return = 0
05/14/2015 5:45:29.479: P831: Authorizing with Service local-users
05/14/2015 5:45:29.479: P831: Getting User user.1’s UserRecord from UserList Default
05/14/2015 5:45:29.479: P831: No UserRecord found for User user.1 in UserList Default, but none _required_ for Authorization.
05/14/2015 5:45:29.479: P831: Trace of Diameter Packet
05/14/2015 5:45:29.479: P831: Trace of Diameter Packet
05/14/2015 5:45:29.479: P831: Result-Code = Diameter-Authentication-Rejected
05/14/2015 5:45:29.479: P831: Auth-Request-Type = AUTHORIZATION
05/14/2015 5:45:29.479: P831: Origin-Host = 10.81.79.81
05/14/2015 5:45:29.479: P831: Session-Id = .;2096298391;3
05/14/2015 5:45:29.479: P831: Auth-Application-Id = 1
05/14/2015 5:45:29.479: P831: Origin-Realm = cisco.com
05/14/2015 5:45:29.479: P831: Sending response to ar-lnx-vm050.cisco.com
05/14/2015 5:45:29.479: P831: Packet successfully removed
05/14/2015 5:45:29.480: P831: Packet Deleted

Configuring External Authentication Service

See Table 4-17 for more information on how to configure external authentication service.

Configuring the Diameter Application in Prime Access Registrar

For proxying a diameter application message in Prime Access Registrar, ensure that you do the following:
Chapter 8  Diameter

- Configuring the Transport Management Properties
- Registering Applications IDs
- Configuring the Diameter Peers
- Configure the Diameter Service

### Configuring the Transport Management Properties

You need to log into the aregcmd using the CLI interface and configure the Transport Management properties in the `Radius/Advanced/Diameter/`

```plaintext
[ //localhost/Radius/Advanced/Diameter ]
IsDiameterEnabled = TRUE
General/
  Product = CPAR
  Version = 7.0
  AuthApplicationIdList = 1
  AcctApplicationIdList = 3
TransportManagement/
  Identity = 10.77.240.69
  Realm = abc.com
  WatchdogTimeout = 500
  ValidateIncomingMessages = FALSE
  ValidateOutgoingMessages = TRUE
  MaximumNumberOfDiameterPackets = 8194
  ReserveDiameterPacketPool = 0
  DiameterPacketSize = 2048
AdvertisedHostName/
  SCTPOptions/
    MaxInitRetry = 3
    MaxInboundStream = 4
    MaxOutboundStream = 5
    EnableHeartbeat = FALSE
    HeartbeatInterval = 500
```

You need to set the Identity and AdvertisedHostName properties to IP Address or hostname of the machine in which Prime Access Registrar is installed.

```plaintext
-- set Identity 10.77.240.69
Set Identity 10.77.240.69

-- cd AdvertisedHostName
set 1 10.77.240.69
Set the Realm in which Cisco Prime Access Registrar server is present.
-- set Realm cisco.com
Set Realm cisco.com

Save the configuration

-- save

Validating //localhost...
Saving //localhost...
```

```plaintext
ls
[ //localhost/Radius/Advanced/Diameter/TransportManagement ]
  Identity = 10.77.240.69
  Realm = cisco.com
  WatchdogTimeout = 500
  ValidateIncomingMessages = FALSE
Configuring the Diameter Application in Prime Access Registrar

ValidateOutgoingMessages = TRUE
MaximumNumberOfDiameterPackets = 8194
ReserveDiameterPacketPool = 0
DiameterPacketSize = 2048
AdvertisedHostName/
  1. 10.77.240.69
SCTPOptions/
  MaxInitRetry = 3
  MaxInboundStream = 4
  MaxOutboundstream = 5
  EnableHeartbeat = FALSE
  HeartbeatInterval = 500

The description for these properties is available at:
http://www.cisco.com/en/US/docs/net_mgmt/access_registrar/5.1/user/guide/objects.html#wp1145662

Note
Prime Access Registrar can only listen to one port for diameter connections. In the above configuration, the port number is 3868. All of the diameter clients must use this port number to communicate with the Prime Access Registrar.

Registering Applications IDs

You need to register the applications IDs for which Prime Access Registrar needs to route the Diameter Messages.

Registering the Gy application to a diameter stack

To register the Gy application to a diameter stack,

Step 1
Move to the //localhost/Radius/Advanced/Diameter/General directory.

[ //localhost/Radius/Advanced/Diameter ]
IsDiameterEnabled = TRUE
General/
  TransportManagement/

  --> cd General/

[ //localhost/Radius/Advanced/Diameter/General ]
Product = Cisco Prime Access Registrar
Version = 7.0
  AuthApplicationIdList = 1
  AcctApplicationIdList =

For description of these properties, see Diameter Service Properties.

Step 2
Set the AuthApplicationIdList to list of colon separated values of Application IDs.

  --> set AuthApplicationIdList "4"

  Set AuthApplicationIdList 4
Configuring the Diameter Peers

You need to configure the Diameter Peers such as clients and servers in the `/radius/clients` and `/radius/remoteservers` directories. The following is an example for configuring a Diameter client:

```
[ //localhost/Radius/Clients/70dia ]
Name = 70dia
Description =
Protocol = diameter
HostName = 10.81.79.241
PeerPort = 3868
Vendor =
IncomingScript~ =
OutgoingScript~ =
AdvertisedHostName =
AdvertisedRealm =
InitialTimeout = 1000
MaxIncomingRequestRate = 0
WatchDogTimeout = 500
SCTP-Enabled = FALSE
TLS-Enabled = FALSE
```

The following is an example for configuring a Diameter remote server:

```
[ //localhost/Radius/RemoteServers/lap ]
Name = lap
Description =
Protocol = diameter
HostName = 10.77.144.34
Port = 3868
DestinationRealm = cisco.com
ReactivateTimerInterval = 300000
Vendor =
IncomingScript~ =
OutgoingScript~ =
MaxTries = 3
MaxTPSLimit = 0
MaxSessionLimit = 0
InitialTimeout = 2000
LimitOutstandingRequests = FALSE
MaxPendingPackets = 0
MaxOutstandingRequests = 0
DWatchDogTimeout = 2500
SCTP-Enabled = FALSE
TLS-Enabled = FALSE
AdvertiseHostName =
AdvertiseRealm =
```

For description of these properties, see Diameter Service Properties.

**Note**

In order to resolve the hostnames and get the IP addresses, the Prime Access Registrar should either be configured with a DNS server IP, or the client's hostnames and IP addresses should be included in the `/etc/hosts` file.

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1 Prime Access Registrar localhost.localdomain localhost
172.16.29.7 GGSN-Gy
::1 localhost6.localdomain6 localhost6
```
Configure the Diameter Service

To configure the Diameter Service to route the Diameter Messages,

**Step 1** Add a Service of type diameter in /Radius/Services/.

```
--> cd /Radius/Services/
--> add dia-proxy
Added dia-proxy

--> cd dia-proxy

[ //localhost/Radius/Services/dia-proxy ]
 Name = dia-proxy
 Description =
 Type =

--> set Type diameter
Set Type diameter
```

**Step 2** Configure the sticky properties.

```
--> set EnableSticky TRUE
Set EnableSticky TRUE

--> ls

[ //localhost/Radius/Services/dia-proxy ]
 Name = dia-proxy
 Description =
 Type = diameter
IncomingScript~ =
OutgoingScript~ =
EnableSticky = TRUE
StickySessionKey =
StickyCreationCmdList =
StickyDeletionCmdList =
MultiplePeersPolicy = Failover
PeerTimeOutPolicy = Failover
DiaRemoteServers/

--> set StickySessionKey Session-Id#1
Set StickySessionKey Session-Id#1

--> set StickyCreationCmdList 265
Set StickyCreationCmdList 265

--> set StickyDeletionCmdList 275
Set StickyDeletionCmdList 275

--> set MultiplePeersPolicy RoundRobin
```
Set MultiplePeersPolicy RoundRobin

--> ls

[ //localhost/Radius/Services/dia-proxy ]
  Name = dia-proxy  
  Description = 
  Type = diameter  
  IncomingScript~ = 
  OutgoingScript~ =   
  EnableSticky = TRUE 
  StickySessionKey = Session-Id#1 
  StickyCreationCmdList = 265 
  StickyDeletionCmdList = 275 
  MultiplePeersPolicy = RoundRobin 
  PeerTimeOutPolicy = FailOver  
  DiaRemoteServers/

**Step 3** Add the peers to which Prime Access Registrar needs to load balance the diameter messages.

[ //localhost/Radius/RemoteServers/dia1 ]
  Name = dia1  
  Description =  
  Protocol = diameter  
  HostName = 192.168.30.88  
  Port = 3868  
  DestinationRealm =  
  ReactivateTimerInterval = 300000  
  Vendor = 
  IncomingScript~ = 
  OutgoingScript~ = 
  MaxTries = 3  
  MaxTPSLimit = 0  
  MaxSessionLimit = 0  
  InitialTimeout = 2000  
  LimitOutstandingRequests = FALSE  
  MaxPendingPackets = 0  
  MaxOutstandingRequests = 0  
  DWatchDogTimeout = 2500  
  SCTP-Enabled = FALSE  
  TLS-Enabled = FALSE  
  AdvertiseHostName = 
  AdvertiseRealm = 

[ //localhost/Radius/RemoteServers/dia2 ]
  Name = dia2  
  Description =  
  Protocol = diameter  
  HostName =  
  Port = 0  
  DestinationRealm =  
  ReactivateTimerInterval = 300000  
  Vendor = 
  IncomingScript~ = 
  OutgoingScript~ = 
  MaxTries = 3  
  MaxTPSLimit = 0  
  MaxSessionLimit = 0  
  InitialTimeout = 2000  
  LimitOutstandingRequests = FALSE  
  MaxPendingPackets = 0  
  MaxOutstandingRequests = 0  
  DWatchDogTimeout = 2500  
  SCTP-Enabled = FALSE  
  TLS-Enabled = FALSE
AdvertiseHostName = 
AdvertiseRealm = 

--> cd diaRemoteServers/

[ //localhost/Radius/Services/dia-proxy/DiaRemoteServers ]
Entries 0 to 0 from 0 total entries 
Current filter: <all>

--> add dia1
Added dia1

--> cd dia1/

[ //localhost/Radius/Services/dia-proxy/DiaRemoteServers/dia1 ]
Name = dia1
Metric = 2
Weight = 0
IsActive = TRUE

--> cd ..

[ //localhost/Radius/Services/dia-proxy/DiaRemoteServers ]
Entries 1 to 1 from 1 total entries 
Current filter: <all>

dia1/

--> add dia2
Added dia2

--> cd dia2

[ //localhost/Radius/Services/dia-proxy/DiaRemoteServers/dia2 ]
Name = dia2
Metric = 3
Weight = 0
IsActive = TRUE

**Step 4**  Save the configuration details.

--> save
Validating //localhost...
Saving //localhost...

**Step 5**  Set DefaultAuthenticationService and DefaultAuthorizationService in /Radius directory.

--> set DefaultAuthenticationService dia-proxy
Set DefaultAuthenticationService dia-proxy

--> set DefaultAuthorizationService dia-proxy
Set DefaultAuthorizationService dia-proxy

--> save
Validating //localhost...
Saving //localhost...

--> exit
Logging out of localhost...
Step 6  Restart the Prime Access Registrar server.

/cisco-ar/bin/arserver restart

The following illustrates the diameter proxy service configuration which load balances the diameter messages to the remote peers.

```plaintext
[ /Radius/Services/dia-proxy ]
    Name = dia-proxy
    Description =
    Type = diameter
    IncomingScript= =
    OutgoingScript= =
    EnableSticky = TRUE
    StickySessionKey = Session-Id#1
    StickyCreationCmdList = 265
    StickyDeletionCmdList = 275
    MultiplePeersPolicy = RoundRobin
    PeerTimeOutPolicy = FailOver
    DiaRemoteServers/
        Entries 1 to 2 from 2 total entries
        Current filter: <all>
        dia1/
            Name = dia1
            Metric = 2
            Weight = 0
            IsActive = TRUE
        dia2/
            Name = dia2
            Metric = 3
            Weight = 0
            IsActive = TRUE
```

For description of these properties, see Diameter Service Properties.

Writing Diameter Application in Prime Access Registrar

Prime Access Registrar supports extensibility by allowing users to create new:

- authentication/authorization applications
- accounting applications
- command codes
- AVP's

This section contains the following topics:

- Configuring rex script/service for Diameter
- Scripting in Diameter
- Diameter Environment Variables
- Sample rex script/service
- Traces/Logs
Configuring rex script/service for Diameter

To configure script/service for diameter using aregcmd:

**Step 1** Add diameter AVPs in //localhost/Radius/Advanced/DiameterDictionary/DiameterAttributes other than Base stack AVPs.

```
[ //localhost/Radius/Advanced/DiameterDictionary/DiameterAttributes/test ]
Name = test
Description =
Attribute =
VendorID = 0
Mandatory = May
May-Encrypt = No
Protected = May
Type =
```

**Step 2** Write a rex script (C/C++) and add it in the scripting point or rex service.

```
[ //localhost/Radius/Services/diaservice ]
Name = diaservice
Description =
Type = rex
IncomingScript~ =
OutgoingScript~ =
OutagePolicy~ = RejectAll
OutageScript~ =
Filename = librexscript.so
EntryPoint = DiaService
InitEntryPoint =
InitEntryPointArgs =
```

Refer to Sample rex script/service, page 8-17.

---

Scripting in Diameter

Prime Access Registrar supports 'rex' scripts for Diameter protocol. The script can be configured only as the server incoming script. The commands available for scripting are restricted to 'get' and 'put' on the dictionaries. While setting a value to an attribute, the following convention needs to be followed "<type number>,<value>". For example, if a 'Class' attribute needs to be added to the response dictionary with value as "classvalue", then set it as follows in the script:

```
pResponse->put( pResponse, "Class", "1.classvalue", REX_REPLACE );
```

The following is the list of supported scripting types with the respective type numbers:

- AVP_STRING_TYPE = 1
- AVP_ADDRESS_TYPE = 2
- AVP_INTEGER32_TYPE = 3
- AVP_UINT32_TYPE = 4
- AVP_UTF8_STRING_TYPE = 6
- AVP_ENUM_TYPE = 7
- AVP_TIME_TYPE = 11

Setting response attributes via a script is the only mechanism to add authorization attributes for Diameter requests.
Diameter Environment Variables

This section lists the environment variables that you can use in scripts for Diameter messages. Table 8-1 lists the Diameter Environment variables and descriptions.

Table 8-1 Diameter Environment Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request-Type</td>
<td>String value. Get/Set the request and response type for diameter packet.</td>
</tr>
<tr>
<td>Response-Type</td>
<td></td>
</tr>
<tr>
<td>Diameter-Application-Id</td>
<td>String value. Get the application id for the packet. For setting in response, need to use Auth-Application-id AVPs.</td>
</tr>
<tr>
<td>Diameter-Command-Code</td>
<td>String value. Get command code for the diameter packet. It will work only for the access-request packet, not for the accounting request.</td>
</tr>
</tbody>
</table>

Sample rex script/service

```c
int REXAPI DiaService( int iScriptingPoint,
                 rex_AttributeDictionary_t* pRequest,
                 rex_AttributeDictionary_t* pResponse,
                 rex_EnvironmentDictionary_t* pEnviron )
{
    if( iScriptingPoint == REX_START_SERVICE || iScriptingPoint == REX_STOP_SERVICE )
        return REX_OK;
    int iRetVal = REX_ERROR;
    const char* pszRequestType = pEnviron->get( pEnviron, "Request-Type" );
    const char* pszAppId = pEnviron->get( pEnviron, "Diameter-Application-Id" );
    const char* pszCmdCode= pEnviron->get( pEnviron, "Diameter-Command-Code" );
    if(! (pszRequestType && pszAppId && pszCmdCode ))
        return iRetVal;
    // check the request type, Application id and command code
    /*
```
Writing Diameter Application in Prime Access Registrar

Request / Response types
Diameter-Access-Request
Diameter-Access-Accept
Diameter-Access-Reject
Diameter-Accounting-Request
Diameter-Accounting-Response

/*
  if( (strcmp( pszRequestType, "Diameter-Access-Request") == 0) && (strcmp(pszAppId,"1") == 0 ) && (strcmp( pszCmdCode,"265") == 0 ) )
{
  // our application
  // example how to get DiaAttrib from the packet.
  const char* pszSessionId = pRequest ->get( pRequest,"Session-Id",0,0 );
  // print in trace
  if( pszSessionId )
    pEnviron->trace( pEnviron, 5, "Diameter Session Id: %s", pszSessionId );
  // example: how to add dia attrib in response packet
  pResponse->put( pResponse, "Calling-Station-Id", "1,00-01-02-03-05", REX_APPEND );
  pEnviron->put( pEnviron, "Response-Type", "Diameter-Access-Accept" );
  iRetVal = REX_OK;
}
return iRetVal;

Traces/Logs

05/14/2015 6:11:05.796: P79: Packet received from 10.81.79.81
05/14/2015 6:11:05.796: P79: Application id: 1, Cmd code: 265, Flag: 0x80
05/14/2015 6:11:05.796: P79: Using Client: vm050
05/14/2015 6:11:05.796: P79: Packet successfully added
05/14/2015 6:11:05.796: P79: Trace of Diameter Packet
05/14/2015 6:11:05.796: P79: Destination-Realm = cisco.com
05/14/2015 6:11:05.796: P79: User-Name = bob
05/14/2015 6:11:05.796: P79: User-Password = <encrypted>
05/14/2015 6:11:05.796: P79: Auth-Request-Type = AUTHORIZE_ONLY
05/14/2015 6:11:05.796: P79: Origin-Host = ar-lnx-vm050.cisco.com
05/14/2015 6:11:05.796: P79: Session-Id = .:2096298391;2
05/14/2015 6:11:05.796: P79: Auth-Application-Id = 1
05/14/2015 6:11:05.796: P79: Origin-Realm = xyz.com
05/14/2015 6:11:05.796: P79: Tracing the packet after running the rules and policies
05/14/2015 6:11:05.796: P79: Using Client: vm050
05/14/2015 6:11:05.796: P79: FastRule Engine called for packet
05/14/2015 6:11:05.796: P79: Fastrule return = 0
05/14/2015 6:11:05.796: P79: Authorizing with Service DiaService
05/14/2015 6:11:05.796: P79: Rex: environ->get( "Request-Type" ) -> "Diameter-Access-Request"
05/14/2015 6:11:05.797: P79: Rex: environ->get( "Diameter-Application-Id" ) -> "1"
05/14/2015 6:11:05.797: P79: Rex: environ->get( "Diameter-Command-Code" ) -> "265"
05/14/2015 6:11:05.797: P79: Rex: request->get( "Session-Id", 0 ) -> ",:2096298391;2"
05/14/2015 6:11:05.797: P79: Diameter Session Id: ,:2096298391;2
05/14/2015 6:11:05.797: P79: Rex: response->put( "Calling-Station-Id", ,"1,00-01-02-03-05", 0 ) -> TRUE
05/14/2015 6:11:05.797: P79: Rex: environ->put( "Response-Type", "Diameter-Access-Accept" ) -> TRUE
05/14/2015 6:11:05.797: P79: Trace of Diameter Packet
05/14/2015 6:11:05.797: P79: User-Name = bob
05/14/2015 6:11:05.797: P79: Result-Code = Diameter-Success
05/14/2015 6:11:05.797: P79: Auth-Request-Type = AUTHORIZE_ONLY
Prime Access Registrar supports translation of an incoming RADIUS request to a Diameter request and vice versa.

The following services are created to set up the translation framework:

- **Radius-Diameter**—For translation of incoming RADIUS request to Diameter equivalent and then the Diameter response to RADIUS equivalent.
- **Diameter-Radius**—For translation of incoming Diameter request to RADIUS equivalent and then the RADIUS response to Diameter equivalent.

For both the translation services, Prime Access Registrar uses the following scripting points to operate on the original packet and on the newly translated packet based on request and response mapping:

- **PreRequestTranslationScript**—To add/modify/delete incoming RADIUS/Diameter attribute values in the request before translation
- **PostRequestTranslationScript**—To add/modify/delete translated Diameter/RADIUS attributes in the request after translation
- **PreResponseTranslationScript**—To add/modify/delete Diameter/RADIUS attribute values in the response before translation
- **PostResponseTranslationScript**—To add/modify/delete RADIUS/Diameter attributes in the response after translation

RADIUS to Diameter translation comes with 3GPP reverse authorization, if the property is set as True. In that case, the request command mapping must not be defined because the new diameter request is created from the radius request by the 3GPP reverse authorization service. When the diameter response is received from the diameter proxy service, it translates the Diameter response to RADIUS response based on the response mapping configuration and sends radius response to the client.

Both these translation services create and maintain appropriate states (with the necessary identifiers, packet pointers, etc) to correlate Request to Response. The states will be cleared if present beyond the ‘Timeout’ property value and all the retries have been exhausted. You can configure the number of retries under Diameter-RemoteServers.

For more information about the translation parameters, see Simple Services, page 3-23.

**CLI for RADIUS-Diameter Translation**

Following is the CLI for RADIUS to Diameter translation:

```plaintext
[ /Radius/Services/rad-dia ]
   Name = rad-dia
   Description =
   Type = radius-diameter
   DiameterApplicationId = 1
   ProxyServiceName = dia-proxy
   EnableCommandMappings = True
```
TLS Support for Diameter

Prime Access Registrar supports Transport Level Security (TLS) mechanism for Diameter stack. The system provides an option to enable TLS for Diameter client and Diameter remote server. When the TLS option is disabled, communication is established directly using the transport layer without applying any encryption. The Diameter TLS feature uses the CiscoSSL libraries, which are available as part of the Prime Access Registrar package.
Following is the CLI configuration of a Diameter client with TLS support:

```
[ /Radius/Clients/vm31 ]
  Name = vm31
  Description =
  Protocol = diameter
  HostName = ar-lnx-vm031.cisco.com
  PeerPort = 3868
  Vendor =
  IncomingScript~ =
  OutgoingScript~ =
  AdvertisedHostName =
  AdvertisedRealm =
  MaxIncomingRequestRate = 0
  WatchDogTimeout = 500
  SCTP-Enabled = FALSE
  TLS-Enabled = TRUE
  TLSOptions/
    PrivateKeyPassword = cisco
    ServerCertificateFile = /opt/CSCOar/pki/cert.pem
    ServerKeyFile = /opt/CSCOar/pki/key.pem
    CA CertificateFile = /opt/CSCOar/pki/root-cert.pem
    CA CertificatePath =
    PeerVerificationMode = None/Optional/RequireCertificate
    VerificationDepth = 4
    EnableAutoChaining = True
```

Following is the CLI configuration of a Diameter remote server with TLS support:

```
[ /Radius/RemoteServers/vm58 ]
  Name = vm58
  Description =
  Protocol = diameter
  HostName = ar-lnx-vm058.cisco.com
  Port = 4322
  DestinationRealm = cisco.com
  ReactivateTimerInterval = 300000
  Vendor =
  IncomingScript~ =
  OutgoingScript~ =
  MaxTries = 3
  InitialTimeout = 2000
  LimitOutstandingRequests = FALSE
  MaxPendingPackets = 0
  MaxOutstandingRequests = 0
  DWatchDogTimeout = 2500
  SCTP-Enabled = FALSE
  TLS-Enabled = TRUE
  AdvertiseHostName =
  AdvertiseRealm =
  TLSOptions/
    PrivateKeyPassword = cisco
    ServerCertificateFile = /opt/CSCOar/pki/cert.pem
    ServerKeyFile = /opt/CSCOar/pki/key.pem
    CA CertificateFile = /opt/CSCOar/pki/root-cert.pem
    CA CertificatePath =
    PeerVerificationMode = None/Optional/RequireCertificate
    VerificationDepth = 4
    EnableAutoChaining = True
```

For descriptions of the TLS options, see the Network Resources, page 3-108 section of Chapter 3, “Using the Graphical User Interface.”
Managing Diameter Sessions

Diameter provides two kinds of services namely authentication/authorization and accounting only (optional). Diameter sessions can be created when an authentication/authorization request comes from an access point or when an accounting start comes from an access point. When a Diameter client issues an authentication request, Prime Access Registrar sends the packet with a Session-Id AVP, which can be used to correlate a Diameter message with a user-session. When a Session Termination Request (STR) message is received from the Diameter client, Prime Access Registrar releases the sessions. Also Re-authentication requests must be mapped to the corresponding user session. In case of accounting packets, the session is created when the accounting start is received from the Diameter client. The session is deleted when the accounting stop message is received.

Prime Access Registrar creates a new session when it receives an authentication or accounting request packet from a Diameter client and when a user session is not already present. It allocates the resources for the particular session from the resource manager and stores the session in a session backing store. This session backing store is a file where session information is written. When a session termination message or an accounting stop message comes from the Diameter client, the session data is deleted from the backing store. Apart from this, Prime Access Registrar maintains the session state for every session it creates. Session cache will be supported for grouped AVPs.

For more information on session manager and its support for Diameter client, see SessionManagers, page 3-96.
Extensible Authentication Protocols

Cisco Prime Access Registrar (Prime Access Registrar) supports the Extensible Authentication Protocol (EAP) to provide a common protocol for differing authentication mechanisms. EAP enables the dynamic selection of the authentication mechanism at authentication time based on information transmitted in the Access-Request. (This type of EAP authentication mechanism is called an authentication exchange.)

Extensible Authentication Protocols (EAP) provide for support of multiple authentication methods. Cisco Prime Access Registrar supports the following EAP authentication methods:

- EAP-AKA
- EAP-AKA-Prime (EAP-AKA’), page 9-6
- EAP-FAST
- EAP-GTC
- EAP-LEAP
- EAP-MD5
- EAP-Negotiate
- EAP-MSChapV2
- EAP-SIM
- EAP-Transport Level Security (TLS)
- EAP-TTLS
- Protected EAP
  - PEAP Version 0 (Microsoft PEAP)
  - PEAP Version 1 (Cisco PEAP)

In general, you enable each EAP method by creating and configuring a service of the desired type. Use the radclient test tool to confirm that the EAP service has been properly configured and is operational.

Both versions of Protected EAP (PEAP) are able to use other EAP methods as the authentication mechanism that is protected by PEAP encryption. For PEAP Version 0, the supported authentication methods are EAP-MSChapV2, EAP-SIM, EAP-TLS and EAP-Negotiate. For PEAP Version 1, the supported authentication methods are EAP-GTC, EAP-SIM, EAP-TLS and EAP-Negotiate.

The PEAP protocol consists of two phases: an authentication handshake phase and a tunnel phase where another complete EAP authentication exchange takes place protected by the session keys negotiated by phase one. Cisco Prime Access Registrar supports the tunneling of other EAP methods within the PEAP phase two exchange.
Prime Access Registrar supports rolling encryption, which involves generating rolling pseudonym secrets for EAP-SIM, EAP-AKA, and EAP-AKA’ services. For more details, see Rolling Encryption Support for Pseudonym Generation in EAP-SIM, EAP-AKA, and EAP-AKA’ Services, page 9-60.

**EAP-AKA**

Authentication and Key Agreement (AKA) is an EAP mechanism for authentication and session key distribution. It is used in the 3rd generation mobile networks Universal Mobile Telecommunications System (UMTS) and CDMA2000. AKA is based on symmetric keys, and typically runs in a UMTS Subscriber Identity Module (USIM), or a (Removable) User Identity Module ((R) UIM), similar to a smart card. EAP-AKA (Extensible Authentication Protocol Method for UMTS Authentication and Key Agreement) includes optional identity privacy support, optional result indications, and an optional fast reauthentication procedure.

In support of EAP-AKA, the following features are supported:

- support of MAP protocol over SIGTRAN
- support of HLR and/or HSS (3GPP compliant)
- Wx interface
- Support M3UA-SIGTRAN over IP

For more information on Wx Interface Support, see the Wx Interface Support for SubscriberDB Lookup, page 17-48.

Prime Access Registrar server supports migration to a converged IP Next Generation Networks (IP NGN) by supporting SS7 and SIGTRAN (SS7 over IP) for HLR communication to enable the seamlessly transition to next-generation IP-based signaling networks.

Prime Access Registrar supports M3UA-SIGTRAN to fetch the authentication vectors from HLR for EAP-AKA authentication, See SIGTRAN-M3UA for more information.

EAP-AKA is based on rfc-4187 (http://www.ietf.org/rfc/rfc4187.txt). This document specifies the details of the algorithms and messages.

This section contains the following topics:

- Configuring EAP-AKA, page 9-2
- Testing EAP-AKA with radclient, page 9-6

**Configuring EAP-AKA**

You can use aregcmd to create and configure a service of type eap-aka.

Table 9-1 lists and describes the EAP-AKA service properties.
### Table 9-1  EAP-AKA Service Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlwaysRequestIdentity</td>
<td>When True, enables the server to obtain the subscriber's identity via EAP/AKA messages instead of relying on the EAP messages alone. This might be useful in cases where intermediate software layers can modify the identity field of the EAP-Response/Identity message. The default value is False.</td>
</tr>
<tr>
<td>EnableIdentityPrivacy</td>
<td>When True, the identity privacy feature is enabled. The default value is False.</td>
</tr>
<tr>
<td>PseudonymSecret</td>
<td>The secret string that is used as the basis for protecting identities when identity privacy is enabled. This should be at least 16 characters long and have a value that is impossible for an outsider to guess. The default value is secret. This field is not available if the EnableRollingPseudonymSecret field is checked. Note: It is very important to change PseudonymSecret from its default value to a more secure value when identity privacy is enabled for the first time.</td>
</tr>
<tr>
<td>PseudonymRenewtime</td>
<td>Specifies the maximum age a pseudonym can have before it is renewed. When the server receives a valid pseudonym that is older than this, it generates a new pseudonym for that subscriber. The value is specified as a string consisting of pairs of numbers and units, where the units might be of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks. The default value is &quot;24 Hours&quot;. Examples are: &quot;8 Hours&quot;, &quot;10 Hours 30 Minutes&quot;, &quot;5 D 6 H 10 M&quot;</td>
</tr>
<tr>
<td>PseudonymLifetime</td>
<td>Specifies the maximum age a pseudonym can have before it is rejected by the server, forcing the subscriber to authenticate using its permanent identity. The value is specified as a string consisting of pairs of numbers and units, where the units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks. It can also be Forever, in which case, pseudonyms do not have a maximum age. The default value is &quot;Forever&quot;. Examples are: &quot;Forever&quot;, &quot;3 Days 12 Hours 15 Minutes&quot;, &quot;52 Weeks&quot;</td>
</tr>
<tr>
<td>NotificationService</td>
<td>(Optional); Notification service is an authorization service and is used to send a notification code to the client in case of an authorization failure. For more information about the Notification-Code variable, see Appendix B, “Environment Dictionary.” This can be any of the services configured under /radius/services/ except eap services, accounting services, radius-session, radius-query, and diameter.</td>
</tr>
<tr>
<td>EnableReauthentication</td>
<td>When True, the fast reauthentication option is enabled. The default value is False.</td>
</tr>
<tr>
<td>UseOutagePolicyForReauth</td>
<td>Default value is FALSE. When set to TRUE, Prime Access Registrar drops or rejects reauthentication requests as per outage policy when the remote server is down. This can be processed only when there is at least one failed full authentication before proceeding with reauthentication.</td>
</tr>
</tbody>
</table>
Table 9-1  EAP-AKA Service Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaximumReauthentications</td>
<td>Specifies the maximum number of times a reauthentication identity might be reused before it must be renewed. The default value is 16.</td>
</tr>
<tr>
<td>ReauthenticationTimeout</td>
<td>Specifies the time in seconds that reauthentication identities are cached by the server. Subscribers that attempt to reauthenticate using identities that are older than this value will be forced to use full authentication instead. The default value is 3600 (one hour).</td>
</tr>
<tr>
<td>ReauthenticationRealm</td>
<td>Optional. If you configure the realm, this value is appended to the FastReauthenticationUserId.</td>
</tr>
<tr>
<td>AuthenticationTimeout</td>
<td>Time in seconds to wait for authentication to complete. The default is 2 minutes; range is 10 seconds to 10 minutes.</td>
</tr>
<tr>
<td>QuintetGenerationScript~</td>
<td>Optional. If the script is set, the custom scripting point can be used to read the quintets from a flat file or generate quintets instead of fetching the quintets from HLR. If the script is not set, the Prime Access Registrar sends the request to HLR configured in remote server to fetch the quintets.</td>
</tr>
<tr>
<td>UseProtectedResults</td>
<td>Enables or disables the use of protected results messages. Results messages indicate the state of the authentication but are cryptographically protected.</td>
</tr>
<tr>
<td>TripletCacheTimeout</td>
<td>Required; timeout value of triplet cache.</td>
</tr>
<tr>
<td>Subscriber_DBLookup</td>
<td>Required. Must be set to either DIAMETER or SIGTRAN-M3UA. When set to DIAMETER, the HSS lookup happens using the Diameter Wx Interface. You need to configure the DestinationRealm to send the Diameter packets to the RemoteServer. When set to SIGTRAN-M3UA, the HLR/HSS lookup happens using the SIGTRAN protocol. You need to configure the SIGTRAN remote server.</td>
</tr>
<tr>
<td>FetchAuthorizationInfo</td>
<td>Required. When set True, it fetches MSISDN from HLR. This field is displayed when you set Subscriber_DBLookup as SIGTRAN-M3UA.</td>
</tr>
<tr>
<td>IncomingScript~</td>
<td>Optional script Prime Access Registrar server runs when it receives a request from a client for an EAP-AKA/EAP-SIM service.</td>
</tr>
<tr>
<td>OutgoingScript~</td>
<td>Optional script Prime Access Registrar server runs before it sends a response to a client using an EAP-AKA/EAP-SIM service.</td>
</tr>
<tr>
<td>OutageScript~</td>
<td>Optional. If set to the name of a script, Prime Access Registrar runs the script when an outage occurs. This property allows you to create a script that notifies you when the server detects a failure.</td>
</tr>
<tr>
<td>RemoteServers</td>
<td>Remote server which can provide the service.</td>
</tr>
<tr>
<td>EnableRollingPseudonymSecret</td>
<td>Check this box to activate rolling encryption process that involves generating rolling pseudonym secrets for the service. For more information about rolling encryption support, see Rolling Encryption Support for Pseudonym Generation in EAP-SIM, EAP-AKA, and EAP-AKA’ Services, page 9-60.</td>
</tr>
</tbody>
</table>
To enable EAP-AKA authentication:

**Step 1** Launch `aregcmd` and create an EAP-AKA service.

```
Step 1 Launch aregcmd and create an EAP-AKA service.

   cd /Radius/Services
   add eap-aka-service
```

**Step 2** Change directory to the service and set its type to eap-aka.

```
Step 2 Change directory to the service and set its type to eap-aka.

   cd eap-aka-service
   set Type eap-aka
```

The following example shows the default configuration for an EAP-AKA service:

```
[ //localhost/Radius/Services/test ]
Name = test
Description =
Type = eap-aka
AlwaysRequestIdentity = False
EnableIdentityPrivacy = False
EnableRollingPseudonymSecret = false
PseudonymSecret = <encrypted>
PseudonymRenewtime = "24 Hours"
PseudonymLifetime = Forever
NotificationService = local-users
Generate3GPPCompliantPseudonym = False
UseOutagePolicyForReauth = False
EnableReauthentication = False
MaximumReauthentications = 16
ReauthenticationTimeout = 3600
ReauthenticationRealm =
AuthenticationTimeout = 120
QuintetGenerationScript~ =
UseProtectedResults = False
SendReAuthIDInAccept = False
SubscriberDBLookup = SIGTRAN-M3UA
FetchAuthorizationInfo = FALSE
MultipleServersPolicy = Failover
IncomingScript~ =
OutgoingScript~ =
OutageScript~ =
RemoteServers/
```

The following example shows the default configuration for an EAP-AKA Wx service:

```
[ //localhost/Radius/Services/eap-aka-wx ]
Name = eap-aka-wx
Description =
Type = eap-aka
AlwaysRequestIdentity = False
EnableIdentityPrivacy = False
PseudonymSecret = <encrypted>
PseudonymRenewtime = "24 Hours"
PseudonymLifetime = Forever
Generate3GPPCompliantPseudonym = False
EnableReauthentication = False
MaximumReauthentications = 16
ReauthenticationTimeout = 3600
```

ReauthenticationRealm = 
AuthenticationTimeout = 120
QuintetGenerationScript~ = 
UseProtectedResults = False
SendReAuthIDInAccept = False
SubscriberDBLookup = Diameter
DestinationRealm = mpc.com
PreRequestTranslationScript~ =
PostRequestTranslationScript~ =
PreResponseTranslationScript~ =
PostResponseTranslationScript~ =

Testing EAP-AKA with radclient

To test the EAP-AKA service, launch radclient and use the simple_eap_aka_test command. The simple_eap_aka_test command sends an Access-Request for the designated user with the user’s secret key and sequence number.

The response packet should indicate an Access-Accept if authentication was successful. View the response packet to ensure the authentication was successful.

   simple_eap_aka_test bob secret 2

To test from radclient, you have to configure /cisco-ar/conf/imsi.conf file on radius server and reload the server. This file content should have imsi users in the format below:

<username>:<secret>:<sequence number>

For example:

bob:1

EAP-AKA-Prime (EAP-AKA')

EAP-AKA-Prime (EAP-AKA') is a new EAP authentication method, with a small revision to the existing EAP-AKA method. EAP- AKA’ has a new key derivation function, which binds the keys derived within the method to the name of the access network. This limits the effects of compromised access network nodes and keys.

EAP- AKA’ is similar to EAP-AKA in all aspects except the following:

- Key derivation involves an AT_KDF_INPUT attribute, which is mapped to the NetworkName attribute, and an AT_KDF attribute, which takes the key derivation function in the configuration, to ensure that the peer and the server know the name of the access network.
- EAP-AKA’ employs SHA-256 (Secure Hash Algorithm) instead of SHA-1 as used in EAP-AKA, to ensure more security.

EAP-AKA’ is based on rfc-5448 (http://www.ietf.org/rfc/rfc5448.txt). This document specifies the details of the algorithms and messages.

This section contains the following topics:
- Configuring EAP-AKA’, page 9-7
- Testing EAP-AKA’ with radclient, page 9-8
Configuring EAP-AKA’

You can use aregcmd to create and configure a service of type eap-aka-prime. EAP-AKA’ service has the following attribute in addition to the service properties listed in Table 9-1.

- **Property**: NetworkName
  - **Description**: Required. Name of the access network for which the authentication is performed. This attribute is captured to ensure that the peer and the server know the name of the access network for performing the EAP authentication.

To enable EAP-AKA’ authentication:

**Step 1** Launch `aregcmd` and create an EAP-AKA’ service.
```
    cd /Radius/Services
    add eap-aka-prime-service
```

**Step 2** Change directory to the service and set its type to eap-aka-prime.
```
    cd eap-aka-prime-service
    set Type eap-aka-prime
```

The following example shows the default configuration for an EAP-AKA’ service:

```
//localhost/Radius/Services/eap-aka-prime {
    Name = eap-aka-prime
    Description = Type = eap-aka-prime
    AlwaysRequestIdentity = False
    EnableIdentityPrivacy = FALSE
    EnableRollingPseudonymSecret = false
    PseudonymSecret = <encrypted>
    PseudonymRenewtime = "24 Hours"
    PseudonymLifetime = Forever
    NotificationService = local-users
    Generate3GPPCompliantPseudonym = False
    EnableReauthentication = FALSE
    UseOutagePolicyForReauth = False
    MaximumReauthentications = 16
    ReauthenticationTimeout = 3600
    ReauthenticationRealm = NetworkName = WAN
    AuthenticationTimeout = 120
    QuintetGenerationScript~ = aka
    UseProtectedResults = TRUE
    SendReAuthTokenInAccept = False
    Subscriber_DBLookup = sigtran
    MultipleServersPolicy = Failover
    IncomingScript~ = 
    OutgoingScript~ =
    OutageScript~ =
    RemoteServers/
```
Testing EAP-AKA’ with radclient

To test the EAP-AKA’ service, launch radclient and use the simple_eap_aka_prime_test command. The simple_eap_aka_prime_test command sends an Access-Request for the designated user with the user's secret key and sequence number.

The response packet should indicate an Access-Accept if authentication was successful. View the response packet to ensure the authentication was successful.

    simple_eap_aka_prime_test bob secret 2

To test from radclient, you have to configure /cisco-ar/conf/imsi.conf file on radius server and reload the server. This file content should have imsi users in the format below:

    <username>:<secret>:<sequence number>

For example:

    bob:bob:1

EAP-FAST

Cisco Prime Access Registrar supports the EAP-FAST authentication method. EAP-FAST uses the EAP-MSChapV2 method for credential provisioning and EAP-GTC for authentication. Credential provisioning typically occurs only during the client’s initial EAP-FAST authentication. Subsequent authentications rely on the provisioned credential and will usually omit the provisioning step.

EAP-FAST is an authentication protocol designed to address the performance shortcomings of prior TLS-based EAP methods while retaining features such as identity privacy and support for password-based protocols. The EAP-FAST protocol is described by the IETF draft draft-cam-winget-eap-fast-00.txt.

The EAP-FAST credential is known as a Protected Access Credential (PAC) and contains information used to secure the authentication operations. Parts of the PAC are encrypted by the server and are not visible to other entities. Clients are expected to securely store PACs locally for use during authentication.

Configuring EAP-FAST involves creating and configuring the required EAP-MSChapV2 and EAP-GTC services as well as the EAP-FAST service with the appropriate parameters.

You can use the radclient test tool to confirm that the EAP services are properly configured and operational.

This section contains the following topics:

- Configuring EAP-FAST
- EAP-FAST Keystores
- Testing EAP-FAST with radclient
- Parameters Used for Certificate-Based Authentication
- PAC—Credential Export Utility
Configuring EAP-FAST

You can use aregcmd to create and configure a service of type eap-fast.

To enable EAP-FAST:

---

**Step 1**  
Launch aregcmd and create an EAP-FAST service.

```
  cd /Radius/Services
  add eap-fast-service
```

**Step 2**  
Change directory to the service and set its type to eap-fast.

```
  cd eap-fast-service
  set type eap-fast
```

**Step 3**  
Set the AuthorityIdentifier:

```
  set AuthorityIdentifier authority-identifier
```

**Step 4**  
Set the AuthorityInformation:

```
  set AuthorityInformation authority-information
```

**Step 5**  
Set the AuthenticationService:

```
  set AuthenticationService eap-gtc-service
```

**Step 6**  
Set the ProvisionService:

```
  set ProvisionService eap-mschapv2-service
```

---

The follow example shows the default configuration for an EAP-FAST service:

```
[ //localhost/Radius/Services/eap-fast-service ]
Name = eap-fast-service
Description =
Type = eap-fast
IncomingScript- =
OutgoingScript- =
MaximumMessageSize = 1024
PrivateKeyPassword = <encrypted>
ServerCertificateFile = /opt/CSCOar/pki/server-cert.pem
ServerKeyFile = /opt/CSCOar/pki/server-key.pem
CACertificateFile = /opt/CSCOar/pki/root-cert.pem
CACertificatePath = /opt/CSCOar/pki
CRLDistributionURL =
ClientVerificationMode = Optional
VerificationDepth = 4
EnableSessionCache = true
UseECCCertificates = true
SessionTimeout = "5 Minutes"
AuthenticationTimeout = 120
```

Table 9-2 lists and describes the EAP-FAST service properties.
Table 9-2  
**EAP-FAST Service Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IncomingScript</td>
<td>Optional script Prime Access Registrar server runs when it receives a request from a client for EAP-FAST service.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional script Prime Access Registrar server runs before it sends a response to a client using EAP-FAST.</td>
</tr>
<tr>
<td>AuthorityIdentifier</td>
<td>A string that uniquely identifies the credential (PAC) issuer. The client uses this value to select the correct PAC to use with a particular server from the set of PACs it might have stored locally. Ensure that the AuthorityIdentifier is globally unique and that it does not conflict with identifiers used by other EAP-FAST servers or PAC issuers.</td>
</tr>
<tr>
<td>AuthorityInformation</td>
<td>A string that provides a descriptive text for this credential issuer. The value can be displayed to the client for identification purposes and might contain the enterprise or server names.</td>
</tr>
<tr>
<td>MaximumMessageSize</td>
<td>Indicates the maximum length in bytes that a PEAP or EAP-TLS message can have before it is fragmented.</td>
</tr>
<tr>
<td>PrivateKeyPassword</td>
<td>The password used to protect the server’s private key.</td>
</tr>
<tr>
<td>ServerCertificateFile</td>
<td>The full pathname of the file containing the server’s certificate or certificate chain used during the TLS exchange. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The two valid encoding prefixes are PEM and DER. If an encoding prefix is not present, the file is assumed to be in PEM format.</td>
</tr>
<tr>
<td>ServerKeyFile</td>
<td>The full pathname of the file containing the server’s RSA or ECC private key. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The two valid encoding prefixes are “PEM” and “DER”. If an encoding prefix is not present, the file is assumed to be in PEM format. The following example assumes that the subdirectory <code>pki</code> under <code>/cisco-ar</code> contains the server’s certificate file. The file <code>server-key.pem</code> is assumed to be in PEM format. The file extension <code>.pem</code> is not significant.</td>
</tr>
<tr>
<td>CACertificateFile</td>
<td>The full pathname of the file containing trusted CA certificates used for client verification. The file can contain more than one certificate, but all certificates must be in PEM format. DER encoding is not allowed.</td>
</tr>
</tbody>
</table>

```bash
set ServerKeyFile PEM:/cisco-ar/pki/server-key.pem
```
CACertificatePath

The name of a directory containing trusted CA certificates (in PEM format) used for client verification. This parameter is optional, and if it is used there are some special preparations required for the directory it references.

Each certificate file in this directory must contain exactly one certificate in PEM format. The server looks up the certificate files using the MD5 hash value of the certificate’s subject name as a key. The directory must therefore also contain a set of symbolic links each of which points to an actual certificate file. The name of each symbolic link is the hash of the subject name of the certificate.

For example, if a certificate file named `ca-cert.pem` is located in the CACertificatePath directory, and the MD5 hash of the subject name contained in `ca-cert.path.pem` is 1b96dd93, then a symbolic link named 1b96dd93 must point to `ca-cert.pem`.

If there are subject name collisions such as multiple certificates with the same subject name, each link name must be indexed with a numeric extension as in 1b96dd93.0 and 1b96dd93.1.

CRLDistributionURL

Optional. Enter the URL that Prime Access Registrar should use to retrieve the CRL. You can specify a URL that uses HTTP or LDAP. The following is an example for an HTTP URL:

```
//crl.verisign.com/pca1.1.1.crl
```

The following is an example for an LDAP URL:

```
ldap://209.165.200.225:388/CN=development-CA,CN=acs-westcoast2,CN=CDP,CN=Public Key Services,CN=Services,CN=Configuration,DC=cisco,DC=com
```

ClientVerificationMode

Specifies the type of verification used for client certificates. Must be set to one of RequireCertificate, None, or Optional.

- RequireCertificate causes the server to request a client certificate and authentication fails if the client refuses to provide one.
- None will not request a client certificate.
- Optional causes the server to request a client certificate but the client is allowed to refuse to provide one.

VerificationDepth

Specifies the maximum length of the certificate chain used for client verification.

UseECCCertificates

Determines the applicability of the authentication mechanism in SmartGrid Solutions, see the Smart Grid Solution Management, page 17-50 for more information.

When UseECCCertificates is set to True, it can use the ECC, RSA, or combination of both certificate for certificate based verification.

When UseECCCertificates is set to False, it can only use the RSA certificate for certificate based verification. The default location to fetch the certificate file is `/cisco-ar/pki`.

---

**Table 9-2 EAP-FAST Service Properties (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CACertificatePath</td>
<td>The name of a directory containing trusted CA certificates (in PEM format) used for client verification. This parameter is optional, and if it is used there are some special preparations required for the directory it references. Each certificate file in this directory must contain exactly one certificate in PEM format. The server looks up the certificate files using the MD5 hash value of the certificate’s subject name as a key. The directory must therefore also contain a set of symbolic links each of which points to an actual certificate file. The name of each symbolic link is the hash of the subject name of the certificate. For example, if a certificate file named <code>ca-cert.pem</code> is located in the CACertificatePath directory, and the MD5 hash of the subject name contained in <code>ca-cert.path.pem</code> is 1b96dd93, then a symbolic link named 1b96dd93 must point to <code>ca-cert.pem</code>. If there are subject name collisions such as multiple certificates with the same subject name, each link name must be indexed with a numeric extension as in 1b96dd93.0 and 1b96dd93.1.</td>
</tr>
<tr>
<td>CRLDistributionURL</td>
<td>Optional. Enter the URL that Prime Access Registrar should use to retrieve the CRL. You can specify a URL that uses HTTP or LDAP. The following is an example for an HTTP URL: <code>&lt; //crl.verisign.com/pca1.1.1.crl&gt;</code>. The following is an example for an LDAP URL: <code>ldap://209.165.200.225:388/CN=development-CA,CN=acs-westcoast2,CN=CDP,CN=Public Key Services,CN=Services,CN=Configuration,DC=cisco,DC=com</code></td>
</tr>
<tr>
<td>ClientVerificationMode</td>
<td>Specifies the type of verification used for client certificates. Must be set to one of RequireCertificate, None, or Optional. - RequireCertificate causes the server to request a client certificate and authentication fails if the client refuses to provide one. - None will not request a client certificate. - Optional causes the server to request a client certificate but the client is allowed to refuse to provide one.</td>
</tr>
<tr>
<td>VerificationDepth</td>
<td>Specifies the maximum length of the certificate chain used for client verification.</td>
</tr>
<tr>
<td>UseECCCertificates</td>
<td>Determines the applicability of the authentication mechanism in SmartGrid Solutions, see the Smart Grid Solution Management, page 17-50 for more information. When UseECCCertificates is set to True, it can use the ECC, RSA, or combination of both certificate for certificate based verification. When UseECCCertificates is set to False, it can only use the RSA certificate for certificate based verification. The default location to fetch the certificate file is <code>/cisco-ar/pki</code>.</td>
</tr>
</tbody>
</table>
Prime Access Registrar verifies the certificate during the TLS-based authentication. CRL validation is done before accepting a client certificate during the TLS authentication.

### EAP-FAST Keystores

The EAP-FAST service manages a set of keys used to protect the security and integrity of the PACs it issues. The keys are stored in `/Radius/Advanced/KeyStores/EAP-FAST` and are maintained automatically requiring minimal administration. Administrators can specify the maximum number of keys that are stored and the frequency of key updates.

The following is the default KeyStores settings:

---

**Table 9-2 EAP-FAST Service Properties (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnableSessionCache</td>
<td>Specifies whether TLS session caching (fast reconnect) is enabled or not. Set to True to enable session caching; otherwise set to False.</td>
</tr>
</tbody>
</table>
| SessionTimeout                | If TLS session caching (fast reconnect) is enabled, SessionTimeout specifies the maximum lifetime of a TLS session. Expired sessions are removed from the cache and will require a subsequent full authentication. SessionTimeout is specified as a string consisting of pairs of numbers and units, where units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks, as in the following:  
  
  **Set SessionTimeout “1 Hour 45 Minutes”**

| AuthenticationTimeout         | Mandatory; specifies time (in seconds) to wait before an authentication request times out; defaults to 120.                                   |
| CredentialLifetime            | Specifies the maximum lifetime of a Protected Access Credential (PAC). Clients that successfully authenticate with an expired PAC will be reprovisioned with a new PAC. CredentialLifetime is specified as a string consisting of pairs of numbers and units, where units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks. Credentials that never expire should be specified as Forever. |
| AuthenticationService          | Specifies the name of the EAP-GTC service is used for authentication. The named service must have the UseLabels parameter set to True.             |
| ProvisionMode                 | Specifies the TLS mode used for provisioning. Clients only support the default Anonymous mode.                                               |
| ProvisionService              | Specifies the name of the EAP-MSChapV2 service used for provisioning.                                                                        |
| AlwaysAuthenticate            | Indicates whether provisioning should always automatically rollover into authentication without relying on a separate session. Most environments, particularly wireless, will perform better when this parameter is set to True, the default value. |

---
Chapter 9 Extensible Authentication Protocols

EAP-FAST

Table 9-3 defines the KeyStores properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NumberOfKeys</td>
<td>Number (from 1-1024) that specifies the maximum number of keys stored for</td>
</tr>
<tr>
<td></td>
<td>EAP-FAST.</td>
</tr>
<tr>
<td>RolloverPeriod</td>
<td>Specifies the amount of time between key updates.</td>
</tr>
</tbody>
</table>

Testing EAP-FAST with radclient

There are two distinct phases to testing EAP-FAST: provisioning and authentication. In the instructions below, Step 2 and Step 3 test provisioning and Steps 4 and Step 5 test authentication. At least one successful provisioning phase must be completed prior to testing authentication. Testing EAP-FAST with radclient requires that the EAP-MSChapV2 and EAP-GTC services be configured and functional.

The following instructions and examples assume that the AlwaysAuthenticate parameter has been set to False for testing purposes. This permits the provisioning and authentication steps to be tested separately. Most installations will set AlwaysAuthenticate to True for production use, and radclient works with that setting, but might display extra error messages that you can ignore.

To test EAP-FAST using radclient:

Step 1 Start radclient.

cd /cisco-ar/usrbin

./radclient –s

Step 2 Specify the inner provisioning method

tunnel eap-mschapv2

The only allowable method for provisioning is eap-mschapv2.

Step 3 Provision a new PAC:

simple_eap_fast_test user-name password

Step 4 Specify the inner authentication method.

tunnel eap-gtc

The only allowable method for authentication is eap-gtc.

Step 5 Authenticate using the PAC.

simple_eap_fast_test user-name password
The `simple_eap_fast_test` command passes its arguments to the inner authentication mechanism which in turn treats the arguments as a username and a password. The command in Step 3 should result in provisioning a new PAC, and Step 5 should result in successful authentication using that PAC.

**PAC Provisioning**

The following example provisions a PAC for user bob.

```
pac show
```

```
No PAC(s) available to show
```

```
tunnel eap-mschapv2
```

```
PEAP tunnel method is eap-mschapv2
EAP-FAST tunnel method is eap-mschapv2
```

```
simple_eap_fast_test bob bob
```

```
EAP-FAST authentication status:
[0x0e07] TLS authentication succeeded
Response to EAP-FAST message was not an Access-Accept
```

```
pac show
```

```
PAC 1 version 1 (219 bytes)
A-ID : Prime AR
A-ID-Info : Cisco Prime Access Registrar
I-ID : bob
Expires : Never (0)
Key# : 12
TLV 1 : PAC-Key (1) mandatory (32 bytes)
TLV 2 : PAC-Opaque (2) mandatory (120 bytes)
TLV 3 : PAC-Info (9) mandatory (51 bytes)
```

In this example the `simple_eap_fast_test` command indicates that it did not receive an AccessAccept. This is normal because the provisioning step always results in an AccessReject even when a new PAC has been successfully provisioned. The last `pac show` command displayed some status information from the new PAC and is used to verify that provisioning succeeded and authentication can now be tested. The PAC information displayed will vary and depends on how EAP-FAST is configured.

**Authentication**

The following example authenticates user bob (continuing from the PAC Provisioning example).

```
tunnel eap-gtc
```

```
PEAP tunnel method is eap-gtc
EAP-FAST tunnel method is eap-gtc
```

```
simple_eap_fast_test bob bob
```

```
EAP-FAST authentication status :
```

```
```

[0x0e07] TLS authentication succeeded
SUCCESS : Correctly formatted Session Keys received from the server
p01e

In this example, the EAP_FAST authentication using the PAC from the previous provisioning step succeeded. The AccessAccept packet received from Prime Access Registrar can be displayed to confirm that it contains the expected attributes including the MS-MPPE session keys.

Parameters Used for Certificate-Based Authentication

EAP-FAST might optionally use RSA or ECC certificates to securely create the tunnel that is used for PAC provisioning. However, the Cisco client does not support the use of certificates and the following parameters will be ignored and should be left at their default values:

- PrivateKeyPassword
- ServerCertificateFile
- ServerKeyFile
- CACertificateFile
- CACertificatePath
- ClientVerificationMode
- VerificationDepth
- UseECCCertificates
- EnableSessionCache
- SessionTimeout

The parameters for configuring certificate-based operation are identical to those used for PEAP and EAP-TLS.

Table 9-4 describes the parameters used for certificate-based authentication.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuthorityIdentifier</td>
<td>A string that uniquely identifies the credential (PAC) issuer. The client uses this value to select the correct PAC to use with a particular server from the set of PACs it might have stored locally. Care should be taken to ensure that the AuthorityIdentifier is globally unique, that is, is distinct from other PAC issuers.</td>
</tr>
<tr>
<td>AuthorityInformation</td>
<td>A string that provides some descriptive text for this credential issuer. The value can be displayed to the client for identification purposes. It can contain the enterprise and/or server names.</td>
</tr>
<tr>
<td>MaximumMessageSize</td>
<td>Indicates the maximum length in bytes that a EAP-FAST message can have before it is fragmented. If certificates are not used for authentication, fragmentation should not be an issue.</td>
</tr>
<tr>
<td>AuthenticationTimeout</td>
<td>Indicates the maximum number of seconds before an authentication operation times out and is rejected.</td>
</tr>
</tbody>
</table>
This section describes the `radclient` commands you can use to test EAP-FAST.

### eap-trace

Use the `eap-trace` command to display additional client protocol trace information for EAP methods. Level is a number from 1 to 5 inclusively. Level 5 shows detailed hex dumps of all messages, level 4 shows a message trace without hex dumps, and levels 3 and below show status and error information. To turn off trace displays, set the level to 0.

Set the trace level for all EAP methods.

**eap-trace level**

For example, the following command sets the trace level to 4 for all EAP methods.

```
eap-trace 4
```

Set the trace level for the specified EAP method.

**eap-trace method level**

The following example sets the trace level to 5 for EAP-FAST only. The trace level for other EAP methods is not affected.

```
eap-trace eap-fast 5
```

**Note** The `eap-trace` command is for client-side trace information only and is independent of the server trace level that can be set using `aregcmd`.
tunnel

The **tunnel** command is used to specify the inner provisioning and authentication methods for EAP-FAST. The specified EAP method type must agree with the server’s configured methods or authentication will fail.

```command
tunnel  eap-method
```

For EAP-FAST provisioning, the only allowable tunnel method is `eap-mchapv2`. For EAP-FAST authentication, the only allowable tunnel method is `eap-gtc`.

**simple_eap_fast_test**

The arguments are passed to the inner authentication method as its authentication parameters. If a PAC is not present, the tunnel method should be `eap-mschapv2` and provisioning will occur. If a PAC is present, the tunnel method should be `eap-gtc` and authentication will occur.

```command
simple_eap_fast_test  username  password
```

There are also variants for the **simple** test command for other EAP methods as shown in the following examples:

```command
simple_eap_mschapv2_test  bob  bob
simple_eap_gtc_test  bob  bob
```

**pac**

The **pac** command is used display, save, and delete PACs that are received from the server during testing. **radclient** maintains a cache of PACs that it knows about and that can be used for authentication testing. The current PAC cache can be displayed with the **pac show** command. PACs created during a test session can be stored to files with the **pac save** command, and reloaded in another session with the **pac load** command. The contents of the PAC cache are completely deleted with **pac delete**. If the optional parameter cache is included, PACs are also erased from disk.

```command
pac load | save | show { hex } | delete { cache }
```

The **pac show** command displays the currently cached PACs. If the optional parameter `hex` is included, additional detailed information including hex dumps are included in the display output.

```command
pac show { hex }
```

The **pac load** command loads any previously saved PACS from disk into the active cache.

The **pac save** command saves all PACS from the active cache to disk. Any previously existing PACS for the same user will be over-written.

The **pac delete** command deletes all PACS from the active cache. If the optional cache parameter is included then PACS are also erased from disk.

```command
pac delete { cache }
```
PAC—Credential Export Utility

You can manually provision EAP-FAST PACs to clients and avoid the use of the protocol provisioning phase. This might be desirable from a security perspective since the default provisioning protocol uses an anonymous (unauthenticated) method to construct the tunnel used to download the PAC to the client.

Manual provisioning involves exporting a PAC from Prime Access Registrar to a file which is then copied to the client machine and used by the import utility. After a PAC has been manually imported, the client should be able to authenticate via EAP-FAST while bypassing the initial provisioning phase. Care should be taken while storing and transporting PAC files since they contain information that potentially allows a client to authenticate via EAP-FAST.

PACs are exported from Prime Access Registrar via the pac command which is a new utility for this release. (Note that this pac command is a standalone executable which is different from the Radclient pac command.) The pac command has two capabilities:

- Exports a PAC to a file
- Displays information about an existing PAC file

PAC Export

Use the pac export command to create a new PAC file. In the following example, eap-fast is the name of the Prime Access Registrar service configured for EAP-FAST authentication, bob is the name of the user this PAC will be used for, and password is the password used to derive a key for encrypting the resulting file. (This password is not the same as the administrator’s password). The PAC file will be named bob.pac by default. You can use the –f option to give the file a different name.

```
pac –s export eap-fast bob password
```

If you omit the password parameter, a default password will be used.

---

**Note**

Using the default password is strongly discouraged for security reasons.
PAC Display

Use the `pac show` command to display information about a PAC file. In the following example, `bob.pac` is the name of the PAC file and `password` is the password used to decrypt the file contents.

```
pac -s show bob.pac password
```

Syntax Summary

The complete `pac` command syntax is as follows:

```
pac { options } export <service-name> <user-name> <file-password>
pac { options } show <file-name> file-<password>
```

Where:
- `-C <cluster>` — Specifies the cluster to be used.
- `-N <user>` — Specifies the user.
- `-P <user-password>` — Specifies the password to be used.
- `-s` — Logs in using defaults
- `-v` — Enables verbose output
- `-f` — Exports file name (default = `{user-name}.pac`)

EAP-GTC

EAP-GTC, defined in RFC 2284, is a simple method for transmitting a user’s name and password to an authentication server. EAP-GTC should not be used except as an authentication method for PEAP Version 1 because the password is not protected.

This section contains the following topics:
- Configuring EAP-GTC
- Testing EAP-GTC with `radclient`

Configuring EAP-GTC

Table 9-5 lists and describes the EAP-GTC specific properties for EAP-GTC authentication.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserService</td>
<td>Required; name of service that can be used to authenticate using cleartext passwords.</td>
</tr>
</tbody>
</table>
To enable EAP-GTC, use `aregcmd` to create and configure a service of type `eap-gtc`.

**Step 1**
Launch `aregcmd` and create an EAP-GTC service.

```
cd /Radius/Services
add eap-gtc-service
```

**Step 2**
Change directory to the service and set its type to `eap-gtc`.

```
cd eap-gtc-service
set type eap-gtc
```

The following example shows the default configuration for an EAP-GTC service:

```
[ //localhost/Radius/Services/eap-gtc-service ]
Name = eap-gtc
Description =
Type = eap-gtc
IncomingScript~ =
OutgoingScript~ =
AuthenticationTimeout = 120
UserService =
UserPrompt = "Enter password:"
UseLabels = False
```

**Step 3**
Set the service’s `UserService` to `local-users` or another local authentication service that is able to authenticate using clear-text passwords.

```
set UserService local-users
```

**Step 4**
If configuring for EAP-FAST, set the `UseLabels` property to `TRUE`.

---

**Testing EAP-GTC with radclient**

To test the EAP-GTC service, launch `radclient` and use the `simple_eap_gtc_test` command. The `simple_eap_gtc_test` command sends an Access-Request for the designated user with the user’s password.

---

**Table 9-5**  
**EAP-GTC Properties (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserPrompt</td>
<td>Optional string the client might display to the user; default is Enter password:” Use the <code>set</code> command to change the prompt, as in the following: set UserPrompt “Admin Password:”</td>
</tr>
<tr>
<td>UseLabels</td>
<td>Required; must be set to TRUE for EAP-FAST authentication and set to FALSE for PEAP authentication. Set to FALSE by default.</td>
</tr>
</tbody>
</table>
The response packet should indicate an Access-Accept if authentication was successful. View the response packet to ensure the authentication was successful.

```
simple_eap_gtc_test bob bob
Packet: code = Access-Accept, id = 2, length = 104, attributes =
Service-Type = Framed
Framed-Protocol = PPP
Framed-IP-Address = 192.168.0.0
Framed-IP-Netmask = 255.255.255.0
Framed-Routing = None
Framed-MTU = 1500
Framed-Compression = VJ TCP/IP header compression
Framed-IPX-Network = 1
EAP-Message = 03:01:00:04
Ascend-Idle-Limit = 1800
```

### EAP-LEAP

Prime Access Registrar supports the new AAA Cisco-proprietary protocol called Light Extensible Authentication Protocol (LEAP), a proprietary Cisco authentication protocol designed for use in IEEE 802.11 wireless local area network (WLAN) environments. Important features of LEAP include:

- Mutual authentication between the network infrastructure and the user
- Secure derivation of random, user-specific cryptographic session keys
- Compatibility with existing and widespread network authentication mechanisms (e.g., RADIUS)
- Computational speed

**Note**

Prime Access Registrar supports a subset of EAP to support LEAP. This is not a general implementation of EAP for Prime Access Registrar.

The Cisco-Wireless or Lightweight Extensible Authentication Protocol is an EAP authentication mechanism where the user password is hashed based on an MD4 algorithm and verified by a challenge from both client and server.

### Configuring EAP-LEAP

You can use `aregcmd` to create and configure a service of type `eap-leap`. When you create an EAP-LEAP service type, you must also specify a UserService to perform AAA service. The UserService can be any configured authentication service.

To enable EAP-LEAP:

**Step 1**

Launch `aregcmd` and create an EAP-LEAP service.

```
cd /Radius/Services
add eap-leap-service
```
Chapter 9  Extensible Authentication Protocols

EAP-MD5

Cisco Prime Access Registrar supports EAP-MD5, or MD5-Challenge, another EAP authentication exchange. In EAP-MD5 there is a CHAP-like exchange and the password is hashed by a challenge from both client and server to verify the password is correct. After verified correct, the connection proceeds, although the connection is periodically re-challenged (per RFC 1994).

Configuring EAP-MD5

Specify type eap-md5 when you create an EAP-MD5 service. When you create an EAP-MD5 service type, you must also specify a UserService to perform AAA service. The UserService can be any configured authentication service.

You can use aregcmd to create and configure a service of type eap-md5. When you create an EAP-MD5 service type, you must also specify a UserService to perform AAA service. The UserService can be any configured authentication service.

To enable EAP-MD5:

Step 1  Launch aregcmd and create an EAP-LEAP service.

        cd /Radius/Services
        add eap-md5-service

Step 2  Set the service type to eap-md5.

        cd eap-md5-service
        set type eap-md5

        [ //localhost/Radius/Services/eap-md5-service ]
        Name = newone
        Description =
        Type =
        IncomingScript- =
        OutgoingScript- =
        AuthenticationTimeout = 120
        UserService =
EAP-Negotiate

EAP-Negotiate is a special service used to select at runtime the EAP service to be used to authenticate the client. EAP-Negotiate is configured with a list of candidate EAP services that represent the allowable authentication methods in preference order. When an EAP session begins, the EAP-Negotiate service tries the first service in the list. If the client does not support that method, it will respond with an EAP-Nak message which triggers EAP-Negotiate to try the next method on the list until a valid method is found or the list is exhausted in which case authentication fails.

EAP-Negotiate is useful when the client population has deployed a mix of different EAP methods that must be simultaneously supported by Prime Access Registrar. It can be difficult or impossible to reliably distinguish which clients require which methods simply by examining RADIUS attributes or other packet properties. EAP-Negotiate solves this problem by using the method negotiation feature of the EAP protocol. Negotiation can be used to select the primary EAP method used for authentication and also to select the inner method for PEAP.

This section contains the following topics:

- Configuring EAP-Negotiate
- Negotiating PEAP Tunnel Services
- Testing EAP-Negotiate with radclient

Configuring EAP-Negotiate

You may first use aregcmd to create and configure the EAP services that will be used for authentication, then create and configure a service of type eap-negotiate.

To enable EAP-Negotiate:

**Step 1** Launch aregcmd and create an EAP-LEAP service.

```
cd /Radius/Services
add eap-negotiate-service
```

**Step 2** Set the service type to eap-negotiate.

```
set type eap-negotiate
```

```
[ //localhost/Radius/Services/negotiate ]
Name = negotiate
Description =
Type = eap-negotiate
IncomingScript~ =
```
**EAP-MSChapV2**

EAP-MSChapV2 is based on `draft-kamath-pppext-eap-mschapv2-00.txt`, an informational IETF draft document. EAP-MSChapV2 encapsulates the MSChapV2 protocol (specified by RFC 2759) and can be used either as an independent authentication mechanism or as an inner method for PEAP Version 0 (recommended).

This section contains the following topics:

- Configuring EAP-MSChapV2
- Testing EAP-MSChapV2 with radclient

**Configuring EAP-MSChapV2**

To enable EAP-MSChapv2, use `aregcmd` to create and configure a service of type `eap-mschapv2`

**Note**

Not all supplicants support negotiation of the PEAP inner method. EAP-Negotiate can only be used with supplicants that can use EAP-Nak to reject an unsupported inner method.

**Testing EAP-Negotiate with radclient**

You can test EAP-Negotiate using the same `radclient` commands used to test the other EAP services. For example, you can use the commands for testing eap-leap and peap-v1.
cd /Radius/Services
add eap-mschapv2

Note
This example named the service eap-mschapv2, but you can use any valid name for your service.

Step 2
Set the service’s type to eap-mschapv2.

cd eap-mschapv2

set Type eap-mschapv2

[ //localhost/Radius/Services/eap-mschapv2 ]
Name = eap-mschapv2
Description =
Type = eap-mschapv2
IncomingScript~ =
OutgoingScript~ =
AuthenticationTimeout = 120
UserService =
SystemID =

Step 3
Set the service’s UserService to local-users or another local authentication service that is able to authenticate using MSChapV2.

set UserService local-users

Step 4
You might (optionally) set a string for System ID that identifies the sender of the MSChapV2 challenge message, as in the following:

set SystemID system_ID_string

Testing EAP-MSChapV2 with radclient

To test the EAP-MSChapVersion 2 service using radclient:

Step 1
Launch radclient.

Step 2
Use the simple_eap_mschapv2_test command to authenticate using EAP-MSChapV2, as in the following:

    simple_eap_mschapv2_test bob bob

    p006

The simple_eap_mschapv2_test command above sends an Access-Request for user bob with the user’s password. The response packet should indicate an Access-Accept if authentication was successful.

Step 3
View the response packet to ensure the authentication was successful.

    p006

    Packet: code = Access-Accept, id = 4, length = 104, attributes =
Cisco Prime Access Registrar supports EAP-SIMv16. In a GSM network a subscriber is issued a smart card called the subscriber identity module (SIM) that contains a secret key (Ki) and an International Mobile Subscriber Identity (IMSI). The key (Ki) is also stored in the GSM authentication center located with the Home Location Registry (HLR).

An access point uses the Prime Access Registrar RADIUS server to perform EAP-SIM authentication of mobile clients. Prime Access Registrar must obtain authentication information from the HLR. Prime Access Registrar contacts the MAP gateway that performs the MAP protocol over SS7 to the HLR, see SIGTRAN-M3UA for more information.

In support of EAP-SIM, the Wx Interface feature will be supported. For more information on Wx Interface Support, see the Wx Interface Support for SubscriberDB Lookup, page 17-48.

### Configuring EAP-SIM

You can use `aregcmd` to create and configure a service of type `eap-sim`.

Table 9-6 lists and describes the EAP-SIM specific properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlwaysRequestIdentity</td>
<td>When True, enables the server to obtain the subscriber's identity via EAP/AKA messages instead of relying on the EAP messages alone. This might be useful in cases where intermediate software layers can modify the identity field of the EAP-Response/Identity message. The default value is False.</td>
</tr>
<tr>
<td>EnableIdentityPrivacy</td>
<td>When True, the identity privacy feature is enabled. The default value is False.</td>
</tr>
</tbody>
</table>
Table 9-6  EAP-SIM Service Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PseudonymSecret</td>
<td>The secret string that is used as the basis for protecting identities when identity privacy is enabled. This should be at least 16 characters long and have a value that is impossible for an outsider to guess. The default value is secret. This field is not available if the EnableRollingPseudonymSecret field is checked.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong> It is very important to change PseudonymSecret from its default value to a more secure value when identity privacy is enabled for the first time.</td>
</tr>
<tr>
<td>PseudonymRenewtime</td>
<td>Specifies the maximum age a pseudonym can have before it is renewed. When the server receives a valid pseudonym that is older than this, it generates a new pseudonym for that subscriber. The value is specified as a string consisting of pairs of numbers and units, where the units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks. The default value is &quot;24 Hours&quot;. Examples are: &quot;8 Hours&quot;, &quot;10 Hours 30 Minutes&quot;, &quot;5 D 6 H 10 M&quot;</td>
</tr>
<tr>
<td>PseudonymLifetime</td>
<td>Specifies the maximum age a pseudonym can have before it is rejected by the server, forcing the subscriber to authenticate using its permanent identity. The value is specified as a string consisting of pairs of numbers and units, where the units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks. It can also be Forever, in which case, pseudonyms do not have a maximum age. The default value is &quot;Forever&quot;. Examples are: &quot;Forever&quot;, &quot;3 Days 12 Hours 15 Minutes&quot;, &quot;52 Weeks&quot;</td>
</tr>
<tr>
<td>NotificationService</td>
<td>(Optional); Notification service is an authorization service and is used to send a notification code to the client in case of an authorization failure. For more information about the Notification-Code variable, see Appendix B, &quot;Environment Dictionary.&quot; This can be any of the services configured under /radius/services/ except eap services, accounting services, radius-session, radius-query, and diameter.</td>
</tr>
<tr>
<td>EnableReauthentication</td>
<td>When True, the fast reauthentication option is enabled. The default value is False.</td>
</tr>
<tr>
<td>UseOutagePolicyforReauth</td>
<td>Default value is FALSE. When set to TRUE, Prime Access Registrar drops or rejects reauthentication requests as per outage policy when the remote server is down. This can be processed only when there is at least one failed full authentication before proceeding with reauthentication.</td>
</tr>
<tr>
<td>MaximumReauthentications</td>
<td>Specifies the maximum number of times a reauthentication identity might be reused before it must be renewed. The default value is 16.</td>
</tr>
<tr>
<td>ReauthenticationTimeout</td>
<td>Specifies the time in seconds that reauthentication identities are cached by the server. Subscribers that attempt to reauthenticate using identities that are older than this value will be forced to use full authentication instead. The default value is 3600 (one hour).</td>
</tr>
<tr>
<td>ReauthenticationRealm</td>
<td>Optional. If you configure the realm, this value is appended to the FastReauthenticationUserId.</td>
</tr>
</tbody>
</table>
Chapter 9  Extensible Authentication Protocols

EAP-SIM

To enable EAP-SIM authentication using aregcmd:

Step 1  Launch aregcmd and create an EAP-SIM service.

  cd /Radius/Services
  add eap-sim-service

Table 9-6  EAP-SIM Service Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuthenticationTimeout</td>
<td>Time in seconds to wait for authentication to complete. The default is 2 minutes; range is 10 seconds to 10 minutes.</td>
</tr>
<tr>
<td>QuintetGenerationScript~</td>
<td>Optional. If the script is set, the custom scripting point can be used to read the quintets from a flat file or generate quintets instead of fetching the quintets from HLR. If the script is not set, the Prime Access Registrar sends the request to HLR configured in remote server to fetch the quintets.</td>
</tr>
<tr>
<td>UseProtectedResults</td>
<td>Enables or disables the use of protected results messages. Results messages indicate the state of the authentication but are cryptographically protected.</td>
</tr>
<tr>
<td>TripletCacheTimeout</td>
<td>Required; timeout value of triplet cache.</td>
</tr>
<tr>
<td>SubscriberDBLookup</td>
<td>Required. Must be set to either DIAMETER or SIGTRAN-M3UA. When set to DIAMETER, the HSS lookup happens using the Diameter Wx Interface. You need to configure the DestinationRealm to send the Diameter packets to the RemoteServer. When set to SIGTRAN-M3UA, the HLR/HSS lookup happens using the SIGTRAN protocol. You need to configure the SIGTRAN remote server.</td>
</tr>
<tr>
<td>FetchAuthorizationInfo</td>
<td>Required. When set True, it fetches MSISDN from HLR. This field is displayed when you set Subscriber_DBLookup as SIGTRAN-M3UA.</td>
</tr>
<tr>
<td>IncomingScript~</td>
<td>Optional script Prime Access Registrar server runs when it receives a request from a client for an EAP-AKA/EAP-SIM service.</td>
</tr>
<tr>
<td>OutgoingScript~</td>
<td>Optional script Prime Access Registrar server runs before it sends a response to a client using an EAP-AKA/EAP-SIM service.</td>
</tr>
<tr>
<td>OutageScript~</td>
<td>Optional. If set to the name of a script, Prime Access Registrar runs the script when an outage occurs. This property allows you to create a script that notifies you when the server detects a failure.</td>
</tr>
<tr>
<td>RemoteServers</td>
<td>Remote server which can provide the service.</td>
</tr>
<tr>
<td>EnableRollingPseudonymSecret</td>
<td>Check this box to activate rolling encryption process that involves generating rolling pseudonym secrets for the service. For more information about rolling encryption support, see Rolling Encryption Support for Pseudonym Generation in EAP-SIM, EAP-AKA, and EAP-AKA’ Services, page 9-60.</td>
</tr>
</tbody>
</table>
Step 2  Change directory to the service and set its type to eap-sim.

    cd eap-sim-service

    set Type eap-sim

    [ //localhost/Radius/Services/EAP-SIM ]
    Name = EAP-SIM
    Description =
    Type = eap-sim
    NumberOfTriplets = 2
    UseSimDemoTriplets = False
    AlwaysRequestIdentity = False
    EnableIdentityPrivacy = False
    EnableRollingPseudonymSecret = false
    PseudonymSecret = <encrypted>
    PseudonymRenewtime = "24 Hours"
    PseudonymLifetime = Forever
    NotificationService = local-users
    Generate3GPPCompliantPseudonym = False
    EnableReauthentication = False
    UseOutagePolicyForReauth = False
    MaximumReauthentications = 16
    ReauthenticationTimeout = 3600
    ReauthenticationRealm =
    TripletCacheTimeout = 120
    AuthenticationTimeout = 120
    UseProtectedResults = False
    SendReAuthIDDInAccept = False
    SubscriberDBLookup = SIGTRAN-M3UA
    FetchAuthorizationInfo = FALSE
    MultipleServersPolicy = Failover
    IncomingScript- =
    OutgoingScript- =
    OutageScript- =
    RemoteServers/

    [ //localhost/Radius/Services/eap-sim-wx ]
    Name = eap-sim-wx
    Description =
    Type = eap-sim
    NumberOfTriplets = 2
    UseSimDemoTriplets = False
    AlwaysRequestIdentity = False
    EnableIdentityPrivacy = False
    PseudonymSecret = <encrypted>
    PseudonymRenewtime = "24 Hours"
    PseudonymLifetime = Forever
    Generate3GPPCompliantPseudonym = False
    EnableReauthentication = False
    MaximumReauthentications = 16
    ReauthenticationTimeout = 3600
    ReauthenticationRealm =
    TripletCacheTimeout = 120
    AuthenticationTimeout = 120
    UseProtectedResults = False
    SendReAuthIDDInAccept = False
    SubscriberDBLookup = DIameter
    DestinationRealm = hss.com
    PreRequestTranslationScript- =
    PostRequestTranslationScript- =
    PreResponseTranslationScript- =
    PostResponseTranslationScript- =
The EAP-SIM property OutagePolicy present in earlier versions of Prime Access Registrar is no longer part of the EAP-SIM configuration.

To enable EAP-SIM authentication using **radclient**:

**Step 1**
Create an EAP-SIM service.

**Step 2**
Change directory to the service and set its type to `eap-sim`.

**Step 3**
Execute the below command in radclient to set session keys in the server.

```
  simple_eap_sim_test 987456321123654 secret
```

**Note**
The IMSI number that is stored in HLR is used for EAP-SIM authentication.

**Step 4**
Enter the server name in which the session key is created to view the `eap-sim` service details.

```
p006
```

```
Packet: code = Access-Accept, id = 3, length = 207, attributes =
User-Name = 987456321123654
MS-MPPE-Send-Key =
EAP-Message = 03:02:00:04
Cisco-AVPair = auth-algo-type=eap-sim
MS-MPPE-Recv-Key =
```

**Quintets to Triplets Conversion**

Prime Access Registrar provides a configuration option in EAP-SIM service, which allows conversion of quintets received from a Universal Mobile Telecommunications Service (UMTS) subscriber to triplets. This feature facilitates backward compatibility by allowing to perform EAP-SIM authentication from an EAP-AKA or EAP-AKA’ source.
EAP-Transport Level Security (TLS)

EAP-Transport Level Security (EAP-TLS), described in RFC 2716, is an authentication method designed to mitigate several weaknesses of EAP. EAP-TLS leverages TLS, described in RFC 2246, to achieve certificate-based authentication of the server and (optionally) the client. EAP-TLS provides many of the same benefits as PEAP but differs from it in the lack of support for legacy authentication methods.

This section contains the following topics:
- Configuring EAP-TLS
- Testing EAP-TLS with RSA or ECC Certificate using radclient
- Testing EAP-TLS with Client Certificates

Configuring EAP-TLS

You can use `aregcmd` to create and configure a service of type `eap-tls`. Table 9-7 describes the EAP-TLS configuration properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IncomingScript</td>
<td>Optional script Prime Access Registrar server runs when it receives a request from a client for EAP-TLS service</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional script Prime Access Registrar server runs before it sends a response to a client using EAP-TLS</td>
</tr>
<tr>
<td>MaximumMessageSize</td>
<td>Indicates the maximum length in bytes that a PEAP or EAP-TLS message can have before it is fragmented.</td>
</tr>
<tr>
<td>PrivateKeyPassword</td>
<td>The password used to protect the server’s private key.</td>
</tr>
<tr>
<td>ServerCertificateFile</td>
<td>The full pathname of the file containing the server’s certificate or certificate chain used during the TLS exchange. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The two valid encoding prefixes are PEM and DER. If an encoding prefix is not present, the file is assumed to be in PEM format.</td>
</tr>
<tr>
<td>ServerKeyFile</td>
<td>The full pathname of the file containing the server’s RSA or ECC (remove for Diameter) private key. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The two valid encoding prefixes are “PEM” and “DER”. If an encoding prefix is not present, the file is assumed to be in PEM format. The following example assumes that the subdirectory <code>pki</code> under <code>/cisco-ar</code> contains the server’s certificate file. The file <code>server-key.pem</code> is assumed to be in PEM format. The file extension <code>.pem</code> is not significant.</td>
</tr>
<tr>
<td>CACertificateFile</td>
<td>The full pathname of the file containing trusted CA certificates used for client verification. The file can contain more than one certificate, but all certificates must be in PEM format. DER encoding is not allowed.</td>
</tr>
</tbody>
</table>
**Table 9-7  EAP-TLS Service Properties (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CACertificatePath</td>
<td>The name of a directory containing trusted CA certificates (in PEM format) used for client verification. This parameter is optional, and if it is used there are some special preparations required for the directory it references. Each certificate file in this directory must contain exactly one certificate in PEM format. The server looks up the certificate files using the MD5 hash value of the certificate’s subject name as a key. The directory must therefore also contain a set of symbolic links each of which points to an actual certificate file. The name of each symbolic link is the hash of the subject name of the certificate. For example, if a certificate file named ca-cert.pem is located in the CACertificatePath directory, and the MD5 hash of the subject name contained in ca-cert.path.pem is 1b96dd93, then a symbolic link named 1b96dd93 must point to ca-cert.pem. If there are subject name collisions such as multiple certificates with the same subject name, each link name must be indexed with a numeric extension as in 1b96dd93.0 and 1b96dd93.1.</td>
</tr>
<tr>
<td>CRLDistributionURL</td>
<td>Optional. The URL that Prime Access Registrar should use to retrieve the CRL. You can specify a URL that uses HTTP or LDAP. The following is an example for an HTTP URL: <a href="http://crl.verisign.com/pca1.1.1.crl">http://crl.verisign.com/pca1.1.1.crl</a>. The following is an example for an LDAP URL: ldap://209.165.200.225:388/CN=development-CA,CN=acs-westcoast2,CN=CDP,CN=Public Key Services,CN=Services,CN=Configuration,DC=cisco,DC=com</td>
</tr>
</tbody>
</table>
| ClientVerificationMode    | Specifies the type of verification used for client certificates. Must be set to one of RequireCertificate, None, or Optional.  
  - RequireCertificate causes the server to request a client certificate and authentication fails if the client refuses to provide one.  
  - None will not request a client certificate.  
  - Optional causes the server to request a client certificate but the client is allowed to refuse to provide one. |
| VerificationDepth         | Specifies the maximum length (in bytes?) of the certificate chain used for client verification. |
| UseECCCertificates        | Determines the applicability of the authentication mechanism in SmartGrid Solutions, see the Smart Grid Solution Management, page 17-50 for more information. When UseECCCertificates is set to True, it can use the ECC, RSA, or combination of both certificate for certificate based verification. When UseECCCertificates is set to False, it can only use the RSA certificate for certificate based verification. The default location to fetch the certificate file is /cisco-ar/pki. |
EAP-Transport Level Security (TLS)

To enable EAP-TLS authentication:

**Step 1** Launch `aregcmd` and create an EAP-TLS service.

```
    cd /Radius/Services
    add eap-tls-service
```

**Step 2** Change directory to the service and set its type to eap-tls.

```
    cd eap-tls-service
    set Type eap-tls
```

```
Name = eap-tls-service
Description =
Type = eap-tls
IncomingScript- =
OutgoingScript- =
MaximumMessageSize = 1024
PrivateKeyPassword = <encrypted>
ServerCertificateFile = /opt/CSCOar/pki/server-cert.pem
ServerKeyFile = /opt/CSCOar/pki/server-key.pem
CACertificateFile = /opt/CSCOar/pki/root-cert.pem
CACertificatePath = /opt/CSCOar/pki
CRLDistributionURL =
ClientVerificationMode = Optional
VerificationDepth = 4
EnableSessionCache = true
UseECCertificates = true
SessionTimeout = "5 Minutes"
AuthenticationTimeout = 120
```

Table 9-7 EAP-TLS Service Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnableSessionCache</td>
<td>Specifies whether TLS session caching (fast reconnect) is enabled or not. Set to True to enable session caching; otherwise set to False.</td>
</tr>
<tr>
<td>SessionTimeout</td>
<td>If TLS session caching (fast reconnect) is enabled, SessionTimeout specifies the maximum lifetime of a TLS session. Expired sessions are removed from the cache and will require a subsequent full authentication. SessionTimeout is specified as a string consisting of pairs of numbers and units, where units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks, as in the following: Set SessionTimeout “1 Hour 45 Minutes”</td>
</tr>
<tr>
<td>AuthenticationTimeout</td>
<td>Mandatory; specifies time (in seconds) to wait before an authentication request times out; defaults to 120.</td>
</tr>
<tr>
<td>EnableAutochaining</td>
<td>When set to TRUE, Prime Access Registrar sends its server certificate chain (Server-Cert -&gt; IntermediateCA -&gt; RootCA) while presenting the server certificate to the client for server side authentication. When set to FALSE, Prime Access Registrar sends only the server certificate (Server-Cert) to the client.</td>
</tr>
</tbody>
</table>
Note
Prime Access Registrar verifies the certificate during the TLS-based authentication. CRL validation is done before accepting a client certificate during the TLS authentication.

Testing EAP-TLS with RSA or ECC Certificate using radclient

To test the EAP-TLS service, launch radclient and use the simple_eap_tls_test command, as in the following:

    simple_eap_tls_test arg1

The argument is arbitrary for the simple_eap_tls_test command and can be anything. You can either select RSA or ECC client certificates using this argument.

Testing EAP-TLS with Client Certificates

You can test EAP-TLS using client certificates verified by the server during the TLS exchange. The client certificate file and RSA or ECC key file must reside in /cisco-ar/pki and be named client-cert.pem and client-key.pem respectively. Both files must be in PEM format.

EAP-TTLS

Prime Access Registrar supports the Extensible Authentication Protocol Tunneled TLS (EAP-TTLS). EAP-TTLS is an EAP protocol that extends EAP-TLS. In EAP-TLS, a TLS handshake is used to mutually authenticate a client and server. EAP-TTLS extends this authentication negotiation by using the secure connection established by the TLS handshake to exchange additional information between client and server.

EAP-TTLS leverages TLS (RFC 2246) to achieve certificate-based authentication of the server (and optionally the client) and creation of a secure session that can then be used to authentication the client using a legacy mechanism. EAP-TTLS provides several benefits:

- Industry standard authentication of the server using certificates (TLS)
- Standardized method for session key generation using TLS PRF
- Strong mutual authentication
- Identity privacy
- Fast reconnect using TLS session caching
- EAP message fragmentation
- Secure support for legacy client authentication methods

EAP-TTLS is a two-phase protocol. Phase 1 conducts a complete TLS session and derives the session keys used in Phase 2 to securely tunnel attributes between the server and the client. The attributes tunneled during Phase 2 can be used to perform additional authentication(s) via a number of different mechanisms.
The authentication mechanisms that can be used during Phase 2 include PAP, CHAP, MS-CHAP, MS-CHAPv2, and EAP. If the mechanism is EAP, then several different EAP methods are possible.

The Phase 2 authentication can be performed by the local AAA Server (the same server running EAP-TTLS) or it can be forwarded to another server (known as the home AAA Server). In the latter case, the home server has no involvement in the EAP-TTLS protocol and can be any AAA service that understands the authentication mechanism in use and is able to authenticate the user. It is not necessary for the home server to understand EAP-TTLS.

This section contains the following topics:

- Configuring EAP-TTLS
- Testing EAP-TTLS with radclient

## Configuring EAP-TTLS

Configuring EAP-TTLS involves two major tasks:

1. Configuring the TLS parameters used for Phase 1
2. Selecting the Phase 2 authentication methods and specifying whether authentication is performed locally or forwarded to the home server.

If authentication is forwarded, the configuration must include the identity of the remote home server and its shared secret.

You configure EAP-TTLS using the `aregcmd` CLI to create the appropriate services and specify their parameters. Use the `radclient` test tool to confirm that the services have been properly configured and are operational.

## Creating an EAP-TTLS Service

You can use `aregcmd` to create and configure a service of type `eap-ttls`. Table 9-8 describes the EAP-TTLS configuration properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IncomingScript</td>
<td>Optional script Prime Access Registrar server runs when it receives a request from a client for EAP-TTLS service.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional script Prime Access Registrar server runs before it sends a response to a client using EAP-TTLS.</td>
</tr>
<tr>
<td>MaximumMessageSize</td>
<td>Indicates the maximum length in bytes that a PEAP or EAP-TLS message can have before it is fragmented.</td>
</tr>
<tr>
<td>PrivateKeyPassword</td>
<td>The password used to protect the server’s private key.</td>
</tr>
<tr>
<td>ServerCertificateFile</td>
<td>The full pathname of the file containing the server’s certificate or certificate chain used during the TLS exchange. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The two valid encoding prefixes are PEM and DER. If an encoding prefix is not present, the file is assumed to be in PEM format.</td>
</tr>
</tbody>
</table>
Table 9-8  EAP-TTLS Service Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| ServerKeyFile     | The full pathname of the file containing the server’s RSA or ECC private key. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The two valid encoding prefixes are “PEM” and “DER”. If an encoding prefix is not present, the file is assumed to be in PEM format. The following example assumes that the subdirectory *pki* under */cisco-ar* contains the server’s certificate file. The file *server-key.pem* is assumed to be in PEM format. The file extension *.pem* is not significant.  
  
  ```plaintext
  set ServerKeyFile PEM:/cisco-ar/pki/server-key.pem
  ```
| CACertificateFile | The full pathname of the file containing trusted CA certificates used for client verification. The file can contain more than one certificate, but all certificates must be in PEM format.  
  
  **Note** DER encoding is not allowed. |
| CACertificatePath | The name of a directory containing trusted CA certificates (in PEM format) used for client verification. This parameter is optional, and if used, there are some special preparations required for the directory it references.  
  
  Each certificate file in this directory must contain exactly one certificate in PEM format. The server looks up the certificate files using the MD5 hash value of the certificate’s subject name as a key. The directory must therefore also contain a set of symbolic links each of which points to an actual certificate file. The name of each symbolic link is the hash of the subject name of the certificate.  
  
  For example, if a certificate file named *ca-cert.pem* is located in the CACertificatePath directory, and the MD5 hash of the subject name contained in *ca-cert.path.pem* is 1b96dd93, then a symbolic link named 1b96dd93 must point to *ca-cert.pem*.  
  
  If there are subject name collisions such as multiple certificates with the same subject name, each link name must be indexed with a numeric extension as in 1b96dd93.0 and 1b96dd93.1.  
  
  **See** rehash-ca-certs Utility, page 9-43 for information about how to create the required certificate file hash links.  
| CRLDistributionURL | Optional. The URL that Prime Access Registrar should use to retrieve the CRL. You can specify a URL that uses HTTP or LDAP.  
  
  The following is an example for an HTTP URL:  
  ```plaintext
  <http://crl.verisign.com/pca1.1.1.crl>
  ```
  
  The following is an example for an LDAP URL:  
  ```plaintext
  ldap://209.165.200.225:388/CN=development-CA,CN=acs-westcoast2,CN=CDP,CN=Public Key Services,CN=Services,CN=Configuration,DC=cisco,DC=com
  ```
### Table 9-8  EAP-TTLS Service Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientVerificationMode</td>
<td>Specifies the type of verification used for client certificates. Must be set to one of RequireCertificate, None, or Optional.</td>
</tr>
<tr>
<td></td>
<td>• RequireCertificate causes the server to request a client certificate and authentication fails if the client refuses to provide one.</td>
</tr>
<tr>
<td></td>
<td>• None will not request a client certificate.</td>
</tr>
<tr>
<td></td>
<td>• Optional causes the server to request a client certificate but the client is allowed to refuse to provide one.</td>
</tr>
<tr>
<td>VerificationDepth</td>
<td>Specifies the maximum length of the certificate chain used for client verification.</td>
</tr>
<tr>
<td>UseECCCertificates</td>
<td>Determines the applicability of the authentication mechanism in SmartGrid Solutions, see the Smart Grid Solution Management, page 17-50 for more information.</td>
</tr>
<tr>
<td></td>
<td>When UseECCCertificates is set to True, it can use the ECC, RSA, or combination of both certificate for certificate based verification.</td>
</tr>
<tr>
<td></td>
<td>When UseECCCertificates is set to False, it can only use the RSA certificate for certificate based verification. The default location to fetch the certificate file is /cisco-ar/pki.</td>
</tr>
<tr>
<td>EnableSessionCache</td>
<td>Specifies whether TLS session caching (fast reconnect) is enabled or not. Set to True to enable session caching; otherwise set to False.</td>
</tr>
<tr>
<td>SessionTimeout</td>
<td>If TLS session caching (fast reconnect) is enabled, SessionTimeout specifies the maximum lifetime of a TLS session. Expired sessions are removed from the cache and require a subsequent full authentication.</td>
</tr>
<tr>
<td></td>
<td>SessionTimeout is specified as a string consisting of pairs of numbers and units, where units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks, as in the following:</td>
</tr>
<tr>
<td></td>
<td>Set SessionTimeout “1 Hour 45 Minutes”</td>
</tr>
<tr>
<td>AuthenticationTimeout</td>
<td>Mandatory; specifies time (in seconds) to wait before an authentication request times out. The default is 120.</td>
</tr>
<tr>
<td>AuthenticationService</td>
<td>Mandatory; specifies the authentication service to use to authenticate users. See Configuring an EAP-TTLS Authentication Service, page 9-38 for more information.</td>
</tr>
<tr>
<td>Note</td>
<td>The authentication service must exist before you can save the EAP-TTLS service configuration.</td>
</tr>
</tbody>
</table>
To enable EAP-TTLS authentication:

**Step 1**
Launch `aregcmd` and create an EAP-TTLS service.

```
cd /Radius/Services
add eap-ttls-service
```

**Step 2**
Change directory to the service and set its type to eap-ttls.

```
cd eap-ttls-service
set Type eap-ttls
```

```
[ //localhost/Radius/Services/eap-ttls-service ]
Name = eap-ttls-service
Description =
Type = eap-ttls
IncomingScript- =
OutgoingScript- =
MaximumMessageSize = 1024
PrivateKeyPassword = <encrypted>
ServerCertificateFile = /opt/CSCOar/pki/server-cert.pem
ServerKeyFile = /opt/CSCOar/pki/server-key.pem
CACertificateFile = /opt/CSCOar/pki/root-cert.pem
CACertificatePath = /opt/CSCOar/pki
CRLDistributionURL =
ClientVerificationMode = Optional
VerificationDepth = 4
EnableSessionCache = true
UseECCCertificates = true
SessionTimeout = "5 Minutes"
AuthenticationTimeout = 120
```

**Note**
Prime Access Registrar verifies the certificate during the TLS-based authentication. CRL validation is done before accepting a client certificate during the TLS authentication.

---

**Configuring an EAP-TTLS Authentication Service**

The EAP-TTLS service can authenticate users with either a legacy method such as PAP, CHAP, MSCHAP, or MSCHAPv2 or with an EAP method such as EAP-MSCHAPv2 or EAP-GTC. The authentication can be performed by the local server (the same server running EAP-TTLS) or it can be forwarded to a remote AAA Server (the home server for the user's domain).

This section provides examples of several different ways to configure an EAP-TTLS authentication service. The following examples assume that you are using aregcmd and have already created the EAP-TTLS service.

**Note**
After you make a configuration change, you must save the configuration before it can be used.
Authenticating Local Users with a Legacy Method

You can use a service like the local-users service (created as part of the example configuration) to authenticate users in the local UserList.

```plaintext
set AuthenticationService local-users
```

This service can be used to authenticate using PAP, CHAP, MSCHAP, and MSCHAPv2.

Authenticating Users with EAP-MSChapV2

This example uses a service named eap-mschapv2 for authentication. Attempts to authenticate using any other method than EAP-MSChapV2 (assuming the service type is also eap-mschapv2) will fail.

```plaintext
set AuthenticationService eap-mschapv2
```

Authenticating Users with EAP Negotiate

You can use the EAP-negotiate method to authenticate using more than one EAP type. The following example defines an EAP service named eap-negotiate that can negotiate EAP-MSChapV2 or EAP-GTC then configures an EAP-TTLS service to authenticate using that service.

To configure an EAP-TTLS service to authenticate using eap-negotiate:

**Step 1**

Create a service of type `eap-negotiate`.

```plaintext
cd /Radius/Services
add eap-nego
cd eap-nego
set Type eap-negotiate
set ServiceList “eap-mschapv2 eap-gtc”
```

**Step 2**

Configure the EAP-TTLS Authentication Service.

```plaintext
cd /Radius/Services/eap-ttls
set AuthenticationService eap-nego
```

Authenticating Users with Legacy and EAP Methods

You can configure EAP-TTLS to authenticate using both legacy and EAP methods with a Group service using an OR result rule. A configuration like that shown in the following example first attempts to authenticate with the eap-negotiate service. If that fails, the server attempts to authenticate with the local-users service.
To authenticate with the eap-negotiate service:

### Step 1
Create the Group service

```bash
cd /Radius/Services
add local-or-eap
cd local-or-eap
set Type group
set ResultRule OR
cd GroupServices
add 1 eap-negotiate
add 2 local-users
```

### Step 2
Configure the EAP-TTLS Authentication Service.

```bash
cd /Radius/Services/eap-ttls
set AuthenticationService local-or-eap
```

---

### Authenticating Using a Remote AAA Server

You can configure an EAP-TTLS service to forward authentication to a remote AAA Server known (or the home server). The following configures a RADIUS service to use a remote server, then configures EAP-TTLS to use that service for authentication.

The first step in the following example configures a remote RADIUS server (aaa-remote) with its IP address and the shared secret that it shares with the local server. You might also specify other important parameters such as ports, timeouts, and maximum number of retries. See Services, page 4-12, for information about configuring RADIUS services.

To configure a remote RADIUS server (aaa-remote) with its IP address and a shared secret:

### Step 1
Configure a remote AAA Server.

```bash
cd /Radius/RemoteServers
add aaa-remote
cd aaa-remote
set Protocol Radius
set IPAddress 10.1.2.3
set SharedSecret secret
```
The following step configures a RADIUS service to use the remote server created in the previous step. You might also configure other important parameters such as the failover strategy. See Services, page 4-12, for information about configuring RADIUS services.

Step 2 Configure an AAA service.

```
  cd /Radius/Services
  add home
  cd home
  set Type Radius
  cd RemoteServers
  add 1 aaa-remote
```

Step 3 Configure the EAP-TTLS AuthenticationService:

```
  cd /Radius/Services/eap-ttls
  set AuthenticationService home
```

Other configurations are also possible. For example, a group service can be used to perform some authentications locally and forward others to a remote server.

**Testing EAP-TTLS with radclient**

To test the EAP-TLS service, launch `radclient` and use the `simple_eap_ttls_test` command. The `simple_eap_ttls_test` command has the following syntax:

```
simple_eap_ttls_test identity password { method }
```

Where:

- `identity` is the user’s name.
- `password` is the user’s password
- `method` is one of: PAP, CHAP, MSChap, MSChapV2, or PEAP.

**Note** If the method parameter is EAP, the `tunnel` command must be used to specify the EAP method type.
Testing EAP-TTLS Using Legacy Methods

To authenticate a user using EAP-TTLS with PAP:

**Step 1** Launch `radclient`.
```
cd /cisco-ar/usrbin
./radclient –s
```

**Step 2** Authenticate using EAP-TTLS PAP.
```
simple_eap_ttls_test bob bob pap
```

The following commands show how to test the other valid legacy methods.
```
simple_eap_ttls_test bob bob chap
simple_eap_ttls_test bob bob mschap
simple_eap_ttls_test bob bob mschapv2
```

Testing EAP-TTLS Using EAP Methods

The following example uses EAP-TTLS with EAP-MSChapV2 as the Phase 2 method to authenticate a user named bob whose password is bob (from the example configuration). Issue the `tunnel` command to specify the Phase 2 EAP method, then issue the `simple_eap_ttls_test` command with `eap` as a method type.

To authenticate a user using EAP-TTLS with EAP-MSChapV2 as the Phase 2 method:

**Step 1** Launch `radclient`
```
cd /cisco-ar/usrbin
./radclient –s
```

**Step 2** Authenticate using EAP-TTLS and EAP-MSChapV2.
```
tunnel eap-mschapv2
simple_eap_ttls_test bob bob eap
```

To test with a different EAP method, use the `tunnel` command to specify the method as shown in the following command to specify EAP-TLS.
```
tunnel eap-tls
simple_eap_ttls_test bob bob eap
```
rehash-ca-certs Utility

The **rehash-ca-certs** utility works with the CACertificatePath property and enables you to create the required certificate file hash links (similar to those used with PEAP and EAP-TLS). The **rehash-ca-certs** utility is only used when the server is validating certificates from the client (which is optional and not a common case for EAP-TTLS).

The syntax for the **rehash-ca-certs** utility is:

```bash
rehash-ca-certs [-v] path1 { path2 ... pathn }
```

Each directory path specified on the command line is scanned by the **rehash-ca-certs** utility for filenames with the **pem** extension (such as `ca-cert.pem`) and the appropriate hash link is created as described above. Before creating links, **rehash-ca-certs** first removes all existing links in the directory, so each invocation creates fresh links. The `-v` option enables verbose output.

The following is an example of the **rehash-ca-certs** utility:

```bash
./rehash-ca-certs ../pki
```

The **rehash-ca-certs** utility warns about PEM files that do not contain certificates. On Cisco Prime Access Registrar, intermediate/chained certificates cannot be imported.

To run Prime Access Registrar with PEAP authentication:

1. **Step 1** Add both root and intermediate CA in the directory `/opt/CSCOar/pki` (as configured for CACertificatePath in the service NYU-NetIDs-PEAPService).
2. **Step 2** Change the directory to pki:
   ```bash
cd /opt/CSCOar/pki
```
3. **Step 3** run `/opt/CSCOar/bin/rehash-ca-certs`
4. **Step 4** Stop ARserver and restart.

**radclient Command Reference**

This section provides a summary of the **radclient** commands you can use to test PEAP and EAP-TLS. It contains the following topics:

- **eap-trace**
- **tunnel**
eap-trace

Use the **eap-trace** command to display additional client protocol trace information for EAP methods. Set the level to a number from 1 to 5 inclusively. Level 5 shows detailed hexadecimal dumps of all messages. Level 4 shows a message trace without hexadecimal dumps. Levels 3 and below show status and error information. To turn off trace displays, set the level to 0.

Use **eap-trace level** to set the trace level for all EAP methods. The following example command sets the trace level to 4 for all EAP methods:

```
eap-trace 4
```

Use **eap-trace method level** to set the trace level for the specified EAP method. The following example command sets the trace level to 5 for PEAP Version0 only. The trace level for other EAP methods is not affected.

```
eap-trace peap-v0 5
```

**Note**
The **eap-trace** command is for client-side trace information only and is independent of the server trace level you set using **aregcmd**.

tunnel

Use the **tunnel** command to specify the inner authentication method for PEAP. The specified EAP method type must agree with the server’s configured authentication method or authentication will fail.

```
tunnel eap-method
```

For PEAP Version 0, the allowable tunnel methods are EAP-MSCHAPV2 and EAP-SIM. For PEAP Version 1, the allowable tunnel methods are EAP-GTC and EAP-SIM.

```
simple_eap_mschapv2_test username password
simple_eap_gtc_test username password
simple_eap_peapv0_test arg1 arg2
```

The arguments are passed to the inner authentication method as its authentication parameters. For EAP-MSChapv2 the arguments are username and password; for EAP-SIM they are IMSI and key.

```
simple_eap_peapv1_test arg1 arg2
```

The arguments are passed to the inner authentication method as its authentication parameters. For EAP-GTC the arguments are username and password; for EAP-SIM they are IMSI and key.

```
simple_eap_tls_test arg1
```
Protected EAP

Protected EAP (PEAP) is an authentication method designed to mitigate several weaknesses of EAP. PEAP leverages TLS (RFC 2246) to achieve certificate-based authentication of the server (and optionally the client) and creation of a secure session that can then be used to authenticate the client. PEAP provides several benefits:

- Industry standard authentication of the server using certificates (TLS)
- Standardized method for session key generation using TLS PRF
- Strong mutual authentication
- Identity privacy
- Fast reconnect using TLS session caching
- EAP message fragmentation
- Secure support for legacy client authentication methods

Cisco Prime Access Registrar supports the two major existing variants of PEAP, PEAP Version 0 (Microsoft PEAP) and PEAP Version 1 (Cisco PEAP). PEAP Version 0 is described in IETF drafts, `draft-kamath-pppext-peapv0-00.txt` and `draft-josefsson-pppext-eap-tls-eap-02.txt`. This version of PEAP can use either EAP-MSChapV2 or EAP-SIM as an authentication method. PEAP Version 1 is described by IETF draft `draft-zhou-pppext-peapv1-00.txt`. PEAP Version 1 can use either EAP-GTC or EAP-SIM as an authentication method.

This section contains the following topics:

- PEAP Version 0
- PEAP Version 1

**PEAP Version 0**

This section describes configuring PEAP Version 0 and testing it with `radclient`.

**Configuring PEAP Version 0**

You can use `aregcmd` to create and configure a service of type `peap-v0`. Table 9-9 describes the PEAP service properties for PEAP Version 0.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IncomingScript</td>
<td>Optional script Prime Access Registrar server runs when it receives a request from a client for PEAP-v0 service.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional script Prime Access Registrar server runs before it sends a response to a client using PEAP-v0</td>
</tr>
<tr>
<td>MaximumMessageSize</td>
<td>Indicates the maximum length in bytes that a PEAP or EAP-TLS message can have before it is fragmented.</td>
</tr>
<tr>
<td>PrivateKeyPassword</td>
<td>The password used to protect the server’s private key.</td>
</tr>
</tbody>
</table>
### Table 9-9  PEAP Version 0 Service Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ServerCertificateFile</td>
<td>The full pathname of the file containing the server’s certificate or certificate chain used during the TLS exchange. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The two valid encoding prefixes are PEM and DER. If an encoding prefix is not present, the file is assumed to be in PEM format. The following example assumes that the subdirectory pki under /cisco-ar contains the server’s certificate file. The file server-cert.pem is assumed to be in PEM format; note that the file extension .pem is not significant.</td>
</tr>
<tr>
<td></td>
<td><code>set ServerCertificateFile PEM:/cisco-ar/pki/server-cert.pem</code></td>
</tr>
<tr>
<td>CACertificateFile</td>
<td>The full pathname of the file containing trusted CA certificates used for client verification. The file can contain more than one certificate, but all certificates must be in PEM format. DER encoding is not allowed.</td>
</tr>
<tr>
<td>CACertificatePath</td>
<td>The name of a directory containing trusted CA certificates (in PEM format) used for client verification. This parameter is optional, and if it is used there are some special preparations required for the directory it references. Each certificate file in this directory must contain exactly one certificate in PEM format. The server looks up the certificate files using the MD5 hash value of the certificate’s subject name as a key. The directory must therefore also contain a set of symbolic links each of which points to an actual certificate file. The name of each symbolic link is the hash of the subject name of the certificate. For example, if a certificate file name ca-cert.pem is located in the CACertificatePath directory, and the MD5 hash of the subject name contained in ca-cert.path.pem is 1b96dd93, then a symbolic link named 1b96dd93 must point to the ca-cert.pem file. If there are subject name collisions such as multiple certificates with the same subject name, each link name must be indexed with a numeric extension as in 1b96dd93.0 and 1b96dd93.1.</td>
</tr>
<tr>
<td>CRLDistributionURL</td>
<td>Optional. The URL that Prime Access Registrar should use to retrieve the CRL. You can specify a URL that uses HTTP or LDAP. The following is an example for an HTTP URL: <a href="http://crl.verisign.com/pca1.1.1.crl">http://crl.verisign.com/pca1.1.1.crl</a>. The following is an example for an LDAP URL: ldap://209.165.200.225:388/CN=development-CA,CN=acs-westcoast2,CN=CDP,CN=Public Key Services,CN=Services,CN=Configuration,DC=cisco,DC=com</td>
</tr>
</tbody>
</table>
### Table 9-9 PEAP Version 0 Service Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientVerificationMode</td>
<td>Specifies the type of verification used for client certificates. Must be set to one of RequireCertificate, None, or Optional.</td>
</tr>
<tr>
<td></td>
<td>• RequireCertificate causes the server to request a client certificate and authentication fails if the client refuses to provide one.</td>
</tr>
<tr>
<td></td>
<td>• None will not request a client certificate.</td>
</tr>
<tr>
<td></td>
<td>• Optional causes the server to request a client certificate but the client is allowed to refuse to provide one.</td>
</tr>
<tr>
<td>VerificationDepth</td>
<td>Specifies the maximum length of the certificate chain used for client verification.</td>
</tr>
<tr>
<td>UseECCCertificates</td>
<td>Determines the applicability of the authentication mechanism in SmartGrid Solutions, see the Smart Grid Solution Management, page 17-50 for more information.</td>
</tr>
<tr>
<td></td>
<td>When UseECCCertificates is set to True, it can use the ECC, RSA, or combination of both certificate for certificate based verification.</td>
</tr>
<tr>
<td></td>
<td>When UseECCCertificates is set to False, it can only use the RSA certificate for certificate based verification. The default location to fetch the certificate file is /cisco-ar/pki.</td>
</tr>
<tr>
<td>EnableSessionCache</td>
<td>Specifies whether TLS session caching (fast reconnect) is enabled or not. Set to True to enable session caching; otherwise set to False.</td>
</tr>
<tr>
<td>SessionTimeout</td>
<td>If TLS session caching (fast reconnect) is enabled, SessionTimeout specifies the maximum lifetime of a TLS session. Expired sessions are removed from the cache and will require a subsequent full authentication.</td>
</tr>
<tr>
<td></td>
<td>SessionTimeout is specified as a string consisting of pairs of numbers and units, where units might be one of the following: M, Minute, Minutes, H, Hour, Hours, D, Day, Days, W, Week, Weeks, as in the following:</td>
</tr>
<tr>
<td>AuthenticationTimeout</td>
<td>Mandatory; specifies time (in seconds) to wait before an authentication request times out; defaults to 120.</td>
</tr>
<tr>
<td>TunnelService</td>
<td>Mandatory; must be the name of an existing EAP-MSCHAPv2 or EAP-SIM service for PEAP Version 0.</td>
</tr>
<tr>
<td>EnableWPS</td>
<td>When set to TRUE, enables Windows Provisioning Service (WPS) and provides two other properties, MasterURL and WPSGuestUserProfile. The default value is FALSE.</td>
</tr>
</tbody>
</table>
To enable PEAP Version 0:

**Step 1** Launch **aregcmd** and create a PEAP Version 0 service.

```
   cd /Radius/Services
   add peap-v0-service
```

**Step 2** Set the service’s type to peap-v0.

```
   cd peap-v0-service
   set Type peap-v0
```

```
   //localhost/Radius/Services/peap-v0-service ]
   Name = peap-v0-service
   Description =
   Type = peap-v0
   IncomingScript~ =
   OutgoingScript~ =
   MaximumMessageSize = 1024
   PrivateKeyPassword = <encrypted>
   ServerCertificateFile = /opt/CSCOar/pki/server-cert.pem
   ServerKeyFile = /opt/CSCOar/pki/server-key.pem
   CACertificateFile = /opt/CSCOar/pki/root-cert.pem
   CACertificatePath = /opt/CSCOar/pki
   CRLDistributionURL =
   ClientVerificationMode = Optional
   VerificationDepth = 4
   EnableSessionCache = true
   UseECCCertificates = true
   SessionTimeout = "5 Minutes"
   AuthenticationTimeout = 120
   EnableWPS = FALSE
```

**Step 3** Set the service’s TunnelService property to the name of an existing EAP-MSCHAPv2 or EAP-SIM service.

```
   set TunnelService name_of_EAP-MSCHAPv2_service
```

This profile is used for guests and users whose account has expired. This profile normally contains attributes denoting the VLAN-id of the guest network (which has the provisioning server alone) and might contain IP-Filters that would restrict the access of the guest (to only the provisioning server).

---

### Table 9-9 PEAP Version 0 Service Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MasterURL</td>
<td>When using WPS, specifies the URL of the provisioning server which is</td>
</tr>
<tr>
<td></td>
<td>modified with the appropriate fragment and sent to the client.</td>
</tr>
<tr>
<td>WPSGuestUserProfile</td>
<td>When using WPS, specifies a profile to be used as a guest user profile;</td>
</tr>
<tr>
<td></td>
<td>must be a valid profile under /Radius/Profiles.</td>
</tr>
</tbody>
</table>

**Note:**
- MasterURL: When using WPS, specifies the URL of the provisioning server which is modified with the appropriate fragment and sent to the client.
- WPSGuestUserProfile: When using WPS, specifies a profile to be used as a guest user profile; must be a valid profile under /Radius/Profiles.

This profile is used for guests and users whose account has expired. This profile normally contains attributes denoting the VLAN-id of the guest network (which has the provisioning server alone) and might contain IP-Filters that would restrict the access of the guest (to only the provisioning server).
set TunnelService name_of_EAP-SIM_service

**Note**
Prime Access Registrar verifies the certificate during the TLS-based authentication. CRL validation is done before accepting a client certificate during the TLS authentication.

### Testing PEAP Version 0 with radclient

To test the PEAP Version 0:

1. **Step 1** Launch `radclient`.
2. **Step 2** Specify the inner authentication method, `eap-mschapv2` or `eap-sim`, as in the following.
   
   ```
   tunnel eap-mschapv2
   ```
   
   or
   
   ```
   tunnel eap-sim
   ```
3. **Step 3** Use the `simple_eap_peapv0_test` command to authenticate using PEAP Version 0, as in the following:
   
   ```
   simple_eap_peapv0_test arg1 arg2
   ```

   The `simple_eap_peapv0_test` command passes its arguments to the inner authentication mechanism which treats the arguments as either a username and a password (for `eap-mschapv2`) or as an IMSI and a key (for `eap-sim`).

The following example tests PEAP Version 0 with EAP-MSCHAPV2 as the inner authentication mechanism using username bob and password bob:

```
  tunnel eap-mschapv2

  simple_eap_peapv0_test bob bob
```

The following example tests PEAP Version 0 with EAP-SIM as the inner authentication mechanism using IMSI 1124567891 and key 0112456789ABCDEF:

```
  tunnel eap-sim

  simple_eap_peapv0_test 1124567891 0112456789ABCDEF
```

### Testing PEAP Version 0 with Client Certificates

You can test PEAP Version 0 using client certificates verified by the server during the TLS exchange. The client certificate file and RSA or ECC key file must reside in `/cisco-ar/pki` and be named `client-cert.pem` and `client-key.pem` respectively. Both files must be in PEM format.
### PEAP Version 1

This section describes configuring PEAP Version 1 and testing it with `radclient`.

#### Configuring PEAP Version 1

You can use `aregcmd` to create and configure a service of type `peap-v1`. Table 9-10 describes the PEAP service properties for both PEAP Version 1.

**Table 9-10 PEAP Version 1 Service Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IncomingScript</td>
<td>Optional script Prime Access Registrar server runs when it receives a request from a client for PEAP-v1 service.</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional script Prime Access Registrar server runs before it sends a response to a client using PEAP-v1.</td>
</tr>
<tr>
<td>MaximumMessageSize</td>
<td>Indicates the maximum length in bytes that a PEAP or EAP-TLS message can have before it is fragmented.</td>
</tr>
<tr>
<td>PrivateKeyPassword</td>
<td>The password used to protect the server’s private key.</td>
</tr>
<tr>
<td>ServerCertificateFile</td>
<td>The full pathname of the file containing the server’s certificate or certificate chain used during the TLS exchange. The pathname can be optionally prefixed with a special string that indicates the type of encoding used for the certificate. The two valid encoding prefixes are PEM and DER. If an encoding prefix is not present, the file is assumed to be in PEM format.</td>
</tr>
<tr>
<td>CACertificateFile</td>
<td>The full pathname of the file containing trusted CA certificates used for client verification. The file can contain more than one certificate but all certificates must be in PEM format. DER encoding is not allowed.</td>
</tr>
<tr>
<td>CACertificatePath</td>
<td>The name of a directory containing trusted CA certificates (in PEM format) used for client verification. This parameter is optional, and if it is used there are some special preparations required for the directory it references. Each certificate file in this directory must contain exactly one certificate in PEM format. The server looks up the certificate files using the MD5 hash value of the certificate’s subject name as a key. The directory must therefore also contain a set of symbolic links each of which points to an actual certificate file. The name of each symbolic link is the hash of the subject name of the certificate. For example, if a certificate file named <code>ca-cert.pem</code> is located in the CACertificatePath directory, and the MD5 hash of the subject name contained in <code>ca-cert.path.pem</code> is 1b96dd93, then a symbolic link named 1b96dd93 must point to the <code>ca-cert.pem</code> file. If there are subject name collisions such as multiple certificates with the same subject name, each link name must be indexed with a numeric extension as in 1b96dd93.0 and 1b96dd93.1.</td>
</tr>
</tbody>
</table>
### Table 9-10  
**PEAP Version 1 Service Properties (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRLDistributionURL</td>
<td>Optional. The URL that Prime Access Registrar should use to retrieve the CRL. You can specify a URL that uses HTTP or LDAP.</td>
</tr>
<tr>
<td></td>
<td>The following is an example for an HTTP URL: <a href="http://crl.verisign.com/pca1.1.1.crl">http://crl.verisign.com/pca1.1.1.crl</a>.</td>
</tr>
<tr>
<td></td>
<td>The following is an example for an LDAP URL: ldap://209.165.200.225:388/CN=development-CA,CN=acs-westcoast 2,CN=CDP,CN=Public Key</td>
</tr>
<tr>
<td></td>
<td>Services,CN=Services,CN=Configuration,DC=cisco,DC=com</td>
</tr>
<tr>
<td>ClientVerificationMode</td>
<td>Specifies the type of verification used for client certificates. Must be set to one of RequireCertificate, None, or Optional.</td>
</tr>
<tr>
<td></td>
<td>• RequireCertificate causes the server to request a client certificate and authentication fails if the client refuses to provide one.</td>
</tr>
<tr>
<td></td>
<td>• None will not request a client certificate.</td>
</tr>
<tr>
<td></td>
<td>• Optional causes the server to request a client certificate but the client is allowed to refuse to provide one.</td>
</tr>
<tr>
<td>VerificationDepth</td>
<td>Specifies the maximum length of the certificate chain used for client verification.</td>
</tr>
<tr>
<td>UseECCCertificates</td>
<td>Determines the applicability of the authentication mechanism in SmartGrid Solutions, see the Smart Grid Solution Management, page 17-50 for more information.</td>
</tr>
<tr>
<td></td>
<td>When UseECCCertificates is set to True, it can use the ECC, RSA, or combination of both certificate for certificate based verification.</td>
</tr>
<tr>
<td></td>
<td>When UseECCCertificates is set to False, it can only use the RSA certificate for certificate based verification. The default location to fetch</td>
</tr>
<tr>
<td></td>
<td>the certificate file is /cisco-ar/pki.</td>
</tr>
<tr>
<td>EnableSessionCache</td>
<td>Specifies whether TLS session caching (fast reconnect) is enabled or not. Set to True to enable session caching; otherwise set to False.</td>
</tr>
<tr>
<td>SessionTimeout</td>
<td>If TLS session caching (fast reconnect) is enabled, SessionTimeout specifies the maximum lifetime of a TLS session. Expired sessions are</td>
</tr>
<tr>
<td></td>
<td>removed from the cache and will require a subsequent full authentication.</td>
</tr>
<tr>
<td></td>
<td>SessionTimeout is specified as a string consisting of pairs of numbers and units, where units might be one of the following: M, Minute, Minutes, H,</td>
</tr>
<tr>
<td></td>
<td>Hour, Hours, D, Day, Days, W, Week, Weeks, as in the following:</td>
</tr>
<tr>
<td></td>
<td>Set SessionTimeout “1 Hour 45 Minutes”</td>
</tr>
<tr>
<td>AuthenticationTimeout</td>
<td>Mandatory; specifies time (in seconds) to wait before an authentication request times out; defaults to 120.</td>
</tr>
<tr>
<td>TunnelService</td>
<td>Mandatory; must be the name of an existing EAP-GTC or EAP-SIM service for PEAP Version 0.</td>
</tr>
</tbody>
</table>
To enable PEAP Version 1:

---

**Step 1** Launch *aregcmd* and create a PEAP Version 1 service.

```
   cd /Radius/Services
   add peap-v1-service
```

**Step 2** Set the service’s type to peap-v1.

```
   cd peap-v1-service
   set Type peap-v1
   //localhost/Radius/Services/eap-peap-v1-service
   Name = eap-peap-v1-service
   Description =
   Type = peap-v1
   IncomingScript =
   OutgoingScript =
   MaximumMessageSize = 1024
   PrivateKeyPassword = <encrypted>
   ServerCertificateFile = /opt/CSCOar/pki/server-cert.pem
   ServerKeyFile = /opt/CSCOar/pki/server-key.pem
   CACertificateFile = /opt/CSCOar/pki/root-cert.pem
   CACertificatePath = /opt/CSCOar/pki
   CRLDistributionURL =
   ClientVerificationMode = Optional
   VerificationDepth = 4
   EnableSessionCache = true
   UseECCCertificates = true
   SessionTimeout = "5 Minutes"
   AuthenticationTimeout = 120
```

**Step 3** Set the service’s TunnelService property to the name of an existing EAP-GTC or EAP-SIM service.

```
   set TunnelService name_of_EAP-GTC_service
   or
   set TunnelService name_of_EAP-SIM_service
```

---

**Testing PEAP Version 1 with radclient**

To test the PEAP Version 1:

---

**Step 1** Launch *radclient*.

**Step 2** Specify the inner authentication method, EAP-GTC or EAP-SIM, as in the following.

```
   tunnel eap-gtc
   or
   tunnel eap-sim
```
Step 3 Use the `simple_eap_peapv1_test` command to authenticate using PEAP Version 1, as in the following:

```
simple_eap_peapv1_test arg1 arg2
```

The `simple_eap_peapv1_test` command passes its arguments to the inner authentication mechanism which treats the arguments as either a username and a password (for EAP-GTC) or as an IMSI and a key (for EAP-SIM).

---

**Testing PEAP Version 1 with Client Certificates**

You can test PEAP Version 1 using client certificates verified by the server during the TLS exchange. The client certificate file and RSA or ECC key file must reside in `/cisco-ar/pki` and be named `client-cert.pem` and `client-key.pem` respectively. Both files must be in PEM format.

---

**How to Configure Oracle, Mysql Accounting with the Buffering Option Enabled**

Prime Access Registrar provides support for MySQL to query user records from Oracle database using sql interface and enables you to write accounting records into Oracle database. You can use insert, update, and delete queries to

- add new details into database.
- modify the existing details in the database.
- remove the outdated details from the database.

---

**To Select the SQL Statement in Run Time Accounting**

Prime Access Registrar provides support to query user account details from SQL database and enables you to add, delete, and update accounting details into SQL when using Oracle accounting.

You can execute the following SQL statements to perform various actions:

- Query
- Insert
- Update
- Delete
- Configuring Oracle, Mysql Accounting

---

**Query**

You can query the accounting details from Oracle by referring this service in `/Radius/DefaultAuthenticationService` and in `/Radius/DefaultAuthorization`.

The following example is an SQL statement used for Authentication and Authorization of the subscribed users. You can use the SQL and MarkerList properties statement to query the selected attributes from Oracle.
Insert

You can insert user details into SQL database by Oracle accounting. This service is used by referring the /Radius/DefaultAccountingService or Accounting-Service environment variable.

For instance, you can use the following SQL and MarkerList properties statement to insert the selected attributes:

```
sql1/
Name = sql1
Description =
Type = query
SQL = "select password , username from arusers where username = ?"
ExecutionSequenceNumber = 1
MarkerList = UserName/SQL_CHAR
```

Update

You can easily modify the details in an SQL table with the UPDATE statement.

For example, you can use the following SQL and MarkerList properties statement to update the selected attributes:

```
sql2/
Name = sql2
Description =
Type = update
SQL = "insert into sql_test (username,nas) values (?,?)"
ExecutionSequenceNumber = 1
MarkerList = UserName/SQL_CHAR NAS-Identifier/SQL_CHAR
```

Delete

You can remove the unnecessary records from SQL database using DELETE statement.

For example, you can use the following SQL and MarkerList properties statement to delete the selected attributes:

```
sql/
Name = sql
Description =
Type = delete
SQL = "delete from arusers_acct where username=?"
ExecutionSequenceNumber = 1
MarkerList = UserName/SQL_CHAR
```
Configuring Oracle, Mysql Accounting

The following script describes you how to configure Oracle, Mysql accounting with the buffering option enabled:

```plaintext
[ //localhost/Radius/Services/oracle-acc ]
Name = oracle-acc
Description =
Type = oci-accounting
IncomingScript~ = sql
OutgoingScript~ =
OutagePolicy~ = RejectAll
OutageScript~ =
MultipleServersPolicy = Failover
RemoteServers/

[ //localhost/Radius/Services/oracle-acc/RemoteServers ]
1. oracle-acc

[ //localhost/Radius/RemoteServers/oracle-acc ]
Name = oracle-acc
Description =
Protocol = oci-accounting
ReactivateTimerInterval = 300000
Timeout = 15
DataSourceConnections = 8
ODBCDataSource = oracle
SNMPTrapIP =
SNMPTrapPort = 1521
KeepAliveTimerInterval = 0
BufferAccountingPackets = TRUE
MaximumBufferSize = "10 Megabytes"
NumberOfRetriesForBufferedPacket = 3
BackingStoreEnvironmentVariables =
UseLocalTimeZone = FALSE
AttributeList =
Delimiter =
SQLDefinition/

[ //localhost/Radius/Advanced/ODBCDataSources/oracle ]
Name = oracle
Description =
Type = oracle_oci
UserID = scott
Password = <encrypted>
DataBase = ORCL

[ //localhost/Radius/Scripts/sql ]
Name = sql
Description =
Language = tcl
Filename = sql.tcl
EntryPoint = sqltest
InitEntryPoint =
InitEntryPointArgs =
```

**Script**

The script statements are executed based on the IP address that you specified in the query. Here is a sample script to select the SQL statements.

```plaintext
proc sqltest {request response environ} {
```
How Suffix and Prefix Rules Work with Prime Access Registrar

Prime Access Registrar includes several scripts that you can use with the rules. The following are the most commonly used rules:

- Prefix Rule, See `ExecPrefixRule, page 18-17` for more information
- Suffix Rule, See `ExecSuffixRule, page 18-18` for more information

Configuring Prefix and Suffix Policies

To configure prefix and suffix policies in Prime Access Registrar in order to provide authentication and authorization services for the subscribed users:

---

**Step 1**
Activate the Policy Engine by configuring `SelectPolicy`. This script explains you how to set a suffix and prefix policy in the grouping list.

```
---> cd selectPolicy/

[ //localhost/Radius/Policies/SelectPolicy ]
Name = SelectPolicy
Description =
Grouping = suffixrule&prefixrule
```

**Step 2**
Run the configuration rules for Prefix and Suffix.

**Step 3**
Set Script = `ExecSuffixRule` in the prefix rule configuration.

```
[ //localhost/Radius/Rules ]
Entries 1 to 2 from 2 total entries
Current filter: <all>

prefixrule/
Name = prefixrule
Description =
Type = radius
Script~ = ExecPrefixRule
Attributes/
  Authentication-Service = local-users
  Authorization-Service = local-users
```

```bash
set nas [ $request get NAS-Identifier ]
if { [ string compare $nas 1.1.1.1 ] == 0 } {
  $environ put SQL-Sequence "sql1"
  $environ put BackingStore-Env-Vars "SQL-Sequence"
}
if { [ string compare $nas 1.1.1.2 ] == 0 } {
  $environ put SQL-Sequence "sql2"
  $environ put BackingStore-Env-Vars "SQL-Sequence"
}
if { [ string compare $nas 1.1.1.3 ] == 0 } {
  $environ put SQL-Sequence "sql3"
  $environ put BackingStore-Env-Vars "SQL-Sequence"
}
if { [ string compare $nas 1.1.1.4 ] == 0 } {
  $environ put SQL-Sequence "sql4"
  $environ put BackingStore-Env-Vars "SQL-Sequence"
}
```
CRL Support for Cisco Prime Access Registrar

Prime Access Registrar checks for various certificates for validation purposes in its authentication services. The client sends a certificate along with the access-challenge to Prime Access Registrar. Prime Access Registrar verifies the validity of the certificate and approves the request if the certificate is valid. For certificate validation, Prime Access Registrar uses an advanced verification mechanism, which uses Certificate Revocation Lists (CRLs).

A CRL, which uses the X.509 certification format, is the signed data structure that the certificate authority (CA) issues periodically. It contains a list of the serial numbers and the timestamp of the revoked certificates. These revoked certificates are not valid and Prime Access Registrar rejects any request that comes with these certificates. The CRLs are available in a public repository in Prime Access Registrar.

A certificate can be revoked because of the following reasons:

- Expiration of the validity period.
- Change in the name of the user to whom the certificate is issued.
- Change in the association between the CA and the user.
- Loss of the private key that is associated with the certificate.

Prime Access Registrar uses the Lightweight Dynamic Authentication Protocol (LDAP) and HTTP for validating the certificates using CRL. The CRLDistributionURL in the TLS based EAP authentication services, is used for the CRL support in Prime Access Registrar. When you configure this property, Prime Access Registrar fetches the CRL from the specified URL, at the startup. A background thread in Prime Access Registrar keeps track of these CRLs. When any of the CRLs expires, Prime Access Registrar fetches the latest version of CRL using the specified URL. Each CRL contains the information related to its expiry.

Prime Access Registrar places all the CRLs in a CRL store. It uses these CRLs while it does a TLS authentication for certificate validation. During an authentication service, the certificate verifier in Prime Access Registrar checks for the validity of the certificate against the CRL issued by the CA that signed the certificate. It looks for the serial number of the certificate in the list of revoked certificates in the appropriate CRL. If it finds a match in the CRL, it compares the revocation time that is encoded in the CRL against the current time. If the current time is later than the revocation time, Prime Access Registrar considers the certificate invalid.

This section contains the following topics:

- Configuring Certificate Validation Using CRL
- Using Intermediate Certificates in Prime Access Registrar
Prime Access Registrar uses the \texttt{CRLDistributionURL} property in the following services:
- \texttt{eap-tls}
- \texttt{eap-ttls}
- \texttt{eap-fast}
- \texttt{peap-v0}
- \texttt{peap-v1}

### Configuring Certificate Validation Using CRL

Prime Access Registrar uses the \texttt{CRLDistributionURL} property for the certificate validation using CRLs. The following shows a sample configuration for the certificate verification using CRLs in Prime Access Registrar:

```bash
//localhost/Radius/Services/eap-ttls-service
Name = eap-ttls-service
Description =
Type = eap-ttls
IncomingScript~ =
OutgoingScript~ =
MaximumMessageSize = 1024
PrivateKeyPassword = <encrypted>
ServerCertificateFile = /opt/CSCOar/pki/server-cert.pem
ServerKeyFile = /opt/CSCOar/pki/server-key.pem
CACertificateFile = /opt/CSCOar/pki/root-cert.pem
CACertificatePath = /opt/CSCOar/pki
CRLDistributionURL =
ClientVerificationMode = Optional
VerificationDepth = 4
EnableSessionCache = true
UseECCCertificates = true
SessionTimeout = "5 Minutes"
AuthenticationTimeout = 120
```

Table 9-8 describes the properties in this sample configuration.

### Using Intermediate Certificates in Prime Access Registrar

The \texttt{rehash-ca-certs} utility can be used to import intermediate certificates in Prime Access Registrar. See \texttt{rehash-ca-certs Utility, page 9-43} for information about how to create the required certificate file hash links.

To import intermediate certificates in Prime Access Registrar:

- **Step 1** Copy the Root CA, Intermediate CA of the client to a directory.
- **Step 2** Run \texttt{/opt/CSCOar/bin/rehash-ca-certs \_\_v <\texttt{path of the client certificate store}>}

The utility creates the required hash links to maintain the chain between the Root CA certificate and Intermediate CA certificates.

- **Step 3** Set the \texttt{CACertificateFile} property in EAP service to the path where Root CA Certificate of the client is stored.
Step 4 Restart the Prime Access Registrar server.

The following shows an example to import intermediate certificates in Prime Access Registrar:

Step 1 Copy the Client Root CA and Intermediate CA Certificate in /cisco-ar/certs/wimax/ directory.

```
cp /tmp/wimax_device_root.pem /cisco-ar/certs/wimax/
cp /tmp/wimax_device_root_ca1.pem /cisco-ar/certs/wimax/
```

```
/opt/CSCOar/bin/rehash-ca-certs -v /cisco-ar/certs/wimax/
```

Step 2 Enter in to aregcmd.

```
/opt/CSCOar/bin/aregcmd --s
```

a. Configure the eap service which uses these client certificates.

```
cd Radius/Services/eap-ttls
```

```
//localhost/Radius/Services/eap-ttls-service ]
Name = eap-ttls-service
Description =
IncomingScript~ =
OutgoingScript~ =
MaximumMessageSize = 1024
PrivateKeyPassword = <encrypted>
ServerCertificateFile = /opt/CSCOar/pki/server-cert.pem
ServerKeyFile = /opt/CSCOar/pki/server-key.pem
CACertificateFile = /opt/CSCOar/pki/root-cert.pem
CACertificatePath = /opt/CSCOar/pki
CRLDistributionURL =
ClientVerificationMode = Optional
VerificationDepth = 4
EnableSessionCache = true
UseECCCertificates = true
SessionTimeout = "5 Minutes"
AuthenticationTimeout = 120
```

```
set CACertificateFile PEM:/opt/CSCOar/pki/wimax_device_root.pem
```

Step 3 Save the configuration.

```
save
```

Step 4 Restart the arserver.

```
/opt/CSCOar/bin/arserver restart
```
Rolling Encryption Support for Pseudonym Generation in EAP-SIM, EAP-AKA, and EAP-AKA’ Services

Prime Access Registrar supports rolling encryption which involves generating rolling pseudonym secrets instead of a single pseudonym secret for EAP-SIM, EAP-AKA, and EAP-AKA’ services. This feature involves the following objects:

- User-Defined Keys
- Key Store
- EAP Service

**User-Defined Keys**
Prime Access Registrar allows you to configure pseudonym encryption keys. There are two types of user-defined keys:

- For generating 3GPP compliant pseudonym secrets—configured at 
  /Radius/Advanced/KeyStores/3GPPKeys
- For generating non-3GPP compliant pseudonym secrets—configured at 
  /Radius/Advanced/KeyStores/non3GPPKeys

If user-defined keys are not configured, Prime Access Registrar uses system generated keys for rolling encryption.

**Key Store**
The Key Store maintains a set of keys that are user-defined and/or auto-generated. There is only one active key at any point in time that is used for generating the pseudonym secret. Rest of the keys are inactive and are used for decryption of old pseudonyms if used for pseudonym generation. After a rollover period, another key is selected as an active key and is used for pseudonym generation. Expiry of a pseudonym depends on the PseudonymLifetime property set for the corresponding EAP service.

For each key store, you can configure the number of keys and rollover period for each key, which is one week by default. For each type of EAP service, there are two key stores:

- For generating 3GPP compliant pseudonym secrets—number of keys is limited to 15. Default is 15. Each key should be 16 digits in length.
- For generating non-3GPP compliant pseudonym secrets—number of keys is limited to 1024. Default is 256. Each key should be 29 digits in length.

Following is the CLI configuration of a key store:

```
[ //localhost/Radius/Advanced/KeyStores ]
EAP-FAST/
    NumberOfKeys = 256
    RolloverPeriod = "1 Week"
EAP-SIM/
    NumberOfKeys = 256
    RolloverPeriod = "1 Week"
EAP-SIM-3GPP/
    NumberOfKeys = 15
    RolloverPeriod = "1 Week"
EAP-AKA/
    NumberOfKeys = 256
    RolloverPeriod = "1 Week"
EAP-AKA-3GPP/
    NumberOfKeys = 15
    RolloverPeriod = "1 Week"
```
Rolling Encryption Support for Pseudonym Generation in EAP-SIM, EAP-AKA, and EAP-AKA' Services

EAP-AKAPRIME/
   NumberOfKeys = 256
   RolloverPeriod = "1 Week"
EAP-AKAPRIME-3GPP/
   NumberOfKeys = 15
   RolloverPeriod = "1 Week"
3GPPKeys/
   Non3GPPKeys/

CLI to configure 3GPP keys:

```bash
--> cd 3GPPKeys/
[ //localhost/Radius/Advanced/KeyStores/3GPPKeys ]
--> set 1 erafgageasgaafde
Set 1 erafgageasgaafde
--> set 2 tttttttadfadfaff
Set 2 tttttttadfadfaff
```

CLI to configure non-3GPP keys:

```bash
--> cd ..../Non3GPPKeys/
[ //localhost/Radius/Advanced/KeyStores/Non3GPPKeys ]
--> set 1 sfsgsgghsr
Set 1 sfsgsgghsr
--> set 2
304 Too few arguments
--> set 2 afgfsfgsfsgsf
Set 2 afgfsfgsfsgsf
```

**EAP Service**

Rolling encryption is applicable for an EAP-SIM, EAP-AKA, or EAP-AKA’ service. This feature is applicable only when the parameter EnableRollingPseudonymSecret is set to TRUE for the service. Each service uses either the 3GPP compliant or non-3GPP compliant pseudonym key store based on the Generate3GPPCompliantPseudonym option set for the service.

The EAP service uses the current active key of the corresponding key store to generate the pseudonym. After rollover period of the current active key, the next active key is selected and used for pseudonym generation. When Prime Access Registrar receives the pseudonym back from the client, it decrypts the pseudonym using the same key that was used to generate the pseudonym. Based on the PseudonymRenewtime and PseudonymLifetime set for the service, if the pseudonym received from the client is expired or to be renewed, Prime Access Registrar uses the current active key from the corresponding key store,

**Rolling Encryption Example**

Table 9-11 provides an example of the rolling encryption process.
### Table 9-11  Rolling Encryption Example

| Assumptions |  
| --- | --- |
| EnablesRollingPseudonymSecret and Generate3GPPCompliantPseudonym parameters are set to TRUE. |  
| eap-sim-3gpp key store is used. Current active key in the key store is key x. |  
| Roll over period = 1 week, which means current active key changes every week. |  
| PseudonymRenewtime parameter of the eap service = 2 days, which means the pseudonym is renewed every two days using the current active key. |  
| PseudonymLifetime parameter of the eap service = 5 days, which means pseudonym is expired if is not renewed in five days. |  

| Day 1 | Prime Access Registrar server generates a pseudonym using key x and provides it to the client.  
Client uses this pseudonym in further authentication process. |  
| Day 2 | Client provides the pseudonym back to the Prime Access Registrar server.  
Prime Access Registrar server decrypts it, identifies the original user, and completes the authentication. No changes are done to the pseudonym. |  
| Day 3 | Client contacts Prime Access Registrar server for authentication.  
Prime Access Registrar understands that the pseudonym needs to be renewed and hence generates a new pseudonym using key x and provides it to the client.  
Process continues for Days 4 and 5. |
Table 9-11 Rolling Encryption Example (continued)

<table>
<thead>
<tr>
<th>Day ‘n’</th>
<th>If:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• there is no authentication transaction between the client and the Prime Access Registrar server for the past five days,</td>
</tr>
<tr>
<td></td>
<td>• the client already has a pseudonym provided by the Prime Access Registrar server during its last transaction, and</td>
</tr>
<tr>
<td></td>
<td>• on the ‘n’th day, client contacts Prime Access Registrar server for authentication</td>
</tr>
<tr>
<td></td>
<td>Then, Prime Access Registrar server:</td>
</tr>
<tr>
<td></td>
<td>• understands that the pseudonym is expired,</td>
</tr>
<tr>
<td></td>
<td>• requests for a permanent ID,</td>
</tr>
<tr>
<td></td>
<td>• generates a new pseudonym by using key x, and</td>
</tr>
<tr>
<td></td>
<td>• provides it to the client.</td>
</tr>
</tbody>
</table>

| Day ‘y’ (y = rollover period date +1) | Key x changes to ‘retired’ stage and will be used for decryption of the old pseudonym generated with the old key. |
|                                       | Prime Access Registrar server uses the next active key (key x+1) for pseudonym generation. |
|                                       | Client can continue the authentication process by using the same old pseudonym till its renewal or expiry time irrespective of active key change at the key store. |
|                                       | If the old pseudonym is found to be renewed or expired, Prime Access Registrar server generates the new pseudonym by using key x+1. |
Rolling Encryption Support for Pseudonym Generation in EAP-SIM, EAP-AKA, and EAP-AKA' Services
Using WiMAX in Cisco Prime Access Registrar

Cisco Prime Access Registrar (Prime Access Registrar) supports Worldwide Interoperability for Microwave Access (WiMAX) technology. This feature support in Prime Access Registrar complies with the WiMAX forum NWG_R1_V1.3.1-Stage-3 specifications.

This chapter contains the following sections:

- **WiMAX - An Overview**
- **WiMAX in Cisco Prime Access Registrar**

### WiMAX - An Overview

WiMAX is a standards-based wireless technology that offers high throughput broadband connections over long distances. WiMAX can be used for a number of applications, including “last mile” broadband connections, fixed and mobile cellular service, hotspots and cellular backhaul, and high-speed enterprise connectivity for business. WiMAX is based on the IEEE 802.16d standard for fixed wireless, and the 802.16e standard for mobile wireless. This standard is appealing to customers because it allows mass production of chipsets that reduce CPE costs, ensures multi-vendor interoperability, and reduces investment risk for operators.

The architectural framework of a WiMAX network consists of the Access Service Network (ASN), the Connectivity Service Network (CSN), and a AAA Server. An Access Service Network is a set of network functions that provide radio access to a WiMAX subscriber. The ASN typically provides functions such as network discovery and selection, connectivity service between the MSS and CSN, Radio Resource Management, Multicast and Broadcast Control, Intra-ASN mobility, Paging, and Location Management. The WiMAX architecture consists of both mobile and fixed subscribers, as well as the ASN and CSN.

A CSN is defined as a set of network functions that provide IP connectivity services to the WiMAX subscribers. CSN might comprise network elements such as Routers, Home Agent, AAA proxy/servers, user databases, Policy Servers, Content Service Gateways, Service Selection Gateways, and interworking gateway devices.

The Access Service Network is connected to a home network HCSN (Home Connectivity Service Network) via at least one visited network (Visited Connectivity Service Network VCSN) or intermediate network.

The Visited CSN plays the role of a AAA proxy. During all AAA interaction the VCSN AAA server acts as a RADIUS proxy transporting RADIUS packets between the ASN and the HCSN.
Figure 10-1 describes the network reference model of a typical WiMAX scenario.

![WiMAX Network Reference Model](image)

**WiMAX in Cisco Prime Access Registrar**

Prime Access Registrar uses the Extensible Authentication Protocol (EAP) to enable the WiMAX feature. It also caches the IP attributes and Mobility Keys that are generated during network access authentication. To enable caching of the WiMAX attributes, you must configure the respective resource managers. See Configuring the Resource Manager for WiMAX, page 10-8, for information on configuring resource manager. Figure 10-2 shows the WiMAX workflow in Prime Access Registrar.
The WiMAX workflow in Prime Access Registrar includes:

- Direct interaction between the ASN GW and Prime Access Registrar
- Interaction between the ASN GW and Prime Access Registrar through the HA

This section contains the following topics:

- Direct Interaction Between the ASN GW and Cisco Prime Access Registrar
- Interaction Between ASN GW and Cisco Prime Access Registrar Through HA
- Prepaid and Hot-Lining

Direct Interaction Between the ASN GW and Cisco Prime Access Registrar

When the mobile node (MN) sends a RADIUS request to the ASN GW, it forwards this request to the CSN. If it is VCSN, the VAAA proxies the request with Visited HA address in the Access Request to HAAA. The HAAA initiates an authentication using the EAP service, for example, eap-ttls. The initial Access-Request containing the WiMAX capability and NAS-Port-Type (Type:61) attributes indicate that the specified flow is for a WiMAX request from ASN GW. Prime Access Registrar redirects this request to the WiMAX service that you configure. The WiMAX service redirects the request to the EAP-based Wimax-Authentication-Service for authentication. Upon successful authentication, the WiMAX service redirects the request to Wimax-Session-Manager to allocate the home agent. Subsequently, Prime Access Registrar generates the appropriate keys based on the Extended Master Session Key (EMSK) and records the generated keys in the session cache resource manager as configured, before sending Access-Accept to the ASN GW.
If there is no VCSN, then the HAAA will send the Access-Accept to ASNGW. Otherwise, the HAAA sends the Access-Accept to VAAA. The VAAA then generates the visited HA-RK Key with SPI and Lifetime and sends the access-accept to ASNGW.

The authentication methods followed by Prime Access Registrar are:

- User-only
- Device-only
- Single-EAP Device or User authentication

**Note**

Prime Access Registrar 4.2 does not support Double-EAP authentication.

Prime Access Registrar uses the following values to identify the service-type:

- Framed—for initial authentication
- Authenticate-Only—for reauthentication
- Authorize-Only—for prepaid request

**Note**

Prepaid attributes can also be sent in the initial authentication.

The attributes contained in this flow are listed in Table 10-1. For detailed information on the attributes refer to the WiMAX forum NWG_R1_V1.3.1-Stage-3 specifications document.

**Table 10-1 Attributes: ASN GW-Prime Access Registrar Flow**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User-Name</td>
<td>Must be present. This attribute gets the NAI from the EAP-Response/Identity.</td>
</tr>
<tr>
<td>Service-Type</td>
<td>Must be present and the value is Framed, Authenticate-Only or Authorize-Only.</td>
</tr>
<tr>
<td>WiMAX Capability</td>
<td>This attribute is chosen by the ASN GW. The request to the Prime Access Registrar is provided through the WiMAX-Capability attribute. The server might respond with the chosen WiMAX Capability.</td>
</tr>
<tr>
<td>NAS-Port-Type</td>
<td>The request must contain this attribute with the value 27. This indicates Wireless IEEE 802.16 port when coming from a WiMAX ASN.</td>
</tr>
<tr>
<td>Calling-Station-ID</td>
<td>The request must contain this attribute with the value set to the MAC address of the device in binary format.</td>
</tr>
<tr>
<td>Device-Authentication-Indicator</td>
<td>The request might contain this attribute to indicate whether the device authentication was performed or not and the result of the action.</td>
</tr>
<tr>
<td>CUI</td>
<td>The NAS might intimate the support for CUI by sending the CUI attribute with the value ‘null’.</td>
</tr>
<tr>
<td>GMT-Time-Zone-Offset</td>
<td>The request must contain the offsets in seconds from the GMT at the NAS.</td>
</tr>
</tbody>
</table>
Cisco Prime Access Registrar generates a few more attributes upon successful authentication. These attributes are described in Table 10-2.

### Table 10-1 Attributes: ASN GW-Prime Access Registrar Flow (continued)

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Framed-IP-Address</td>
<td>This is the CMIPv4 Home address to be assigned to the MN. If this attribute is not present then the Home address is derived by the ASN from MIP procedures or through DHCP.</td>
</tr>
<tr>
<td>WiMax-Session-ID</td>
<td>This attribute shall not be present in the initial authentication. The value is a unique identifier in the home realm for this session as set by the HAAA(Prime Access Registrar) in the Access-Accept, when the authentication is successful and it will be included in all subsequent requests from the NAS, such as online accounting.</td>
</tr>
<tr>
<td>MSK</td>
<td>The MSK shall be provided by the AAA Server as a result of successful EAP-authentication. MSK can be transmitted using either the MS-MPPE-Keys or the MSK attribute.</td>
</tr>
<tr>
<td>Packet-Flow-Descriptor</td>
<td>The pre-provisioned service flow which might be present in the Access-Accept packet.</td>
</tr>
<tr>
<td>QoS-Descriptor</td>
<td>The pre-provisioned service flow which might be present in the Access-Accept packet, if configured in Prime Access Registrar.</td>
</tr>
<tr>
<td>BS-ID</td>
<td>Might be present in the Access-Request packet which will identify NAP-ID base station. If both NAP-ID and BS-ID are present, the NAP-ID will be ignored.</td>
</tr>
<tr>
<td>Acct-Interim-Interval</td>
<td>Sent in the Access-Accept packet. It indicates the accounting update intervals.</td>
</tr>
</tbody>
</table>

### Table 10-2 Additional Attributes: ASN-GW Prime Access Registrar Flow

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hHA-IP-MIP4</td>
<td>The IP address of the home HA allocated for the incoming request.</td>
</tr>
<tr>
<td>vHA-IP-MIP4</td>
<td>The IP address of the visited HA. To be used by the PMIP4 client.</td>
</tr>
<tr>
<td>MN-hHA-MIP4-KEY</td>
<td>The MN-hHA key is used for MIP4 procedures.</td>
</tr>
<tr>
<td>MN-hHA-MIP4-SPI</td>
<td>The SPI associated with the MN-hHA-MIP4-KEY.</td>
</tr>
<tr>
<td>MN-vHA-MIP4-KEY</td>
<td>The MN-vHA key is used for MIP4 procedures.</td>
</tr>
</tbody>
</table>
Interaction Between ASN GW and Cisco Prime Access Registrar Through HA

After Prime Access Registrar returns the Access-Accept to the ASN GW, the mobile node, which initially sent the request, sends a registration request to the ASN GW. The ASN GW receives this request and sends an Access-Request to the HA. A Query-Request will be sent to the Prime Access Registrar by HA to receive the security context for authenticating the FA.

Prime Access Registrar identifies the request as HA query request, if:

- the WiMAX mobility attribute is present
- the NAS-Port-Type attribute is absent

Prime Access Registrar checks for a valid session in the session cache based on NAI and sends an Access-Accept to the HA.

### Table 10-3 HAAA Cached Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pseudo Identity</td>
<td>As received from the MS in the NAI in the EAP-Response/Identity. The HAAA is required to correlate this to the true identity of the user.</td>
</tr>
<tr>
<td>NAS-ID/NAS-IP address</td>
<td>One or both of these parameters are cached by the HAAA. This is required to locate the serving NAS.</td>
</tr>
<tr>
<td>Framed-IP Address</td>
<td>The IP address allocated to the user session. This information is useful in identifying the session during AAA dynamic procedures.</td>
</tr>
<tr>
<td>MIP-RK, hHA-RK, FA-RK, MN-hHA</td>
<td>Mobility keys generated during network access authentication. These keys are cached and used by the network for mobility authentication.</td>
</tr>
<tr>
<td>hHA-IP address</td>
<td>The IP address of the home HA assigned to the MS.</td>
</tr>
</tbody>
</table>
Table 10-4 VAAA Cached Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vHA-RK, vHA-RK-SPI, vHA-RK Lifetime, MN-vHA</td>
<td>Mobility keys generated during network access authentication. These keys are cached and used by the network for mobility authentication.</td>
</tr>
<tr>
<td>vHA-IP address</td>
<td>The IP address of the visited HA assigned to the MS.</td>
</tr>
</tbody>
</table>

Note

Prime Access Registrar responds with the correct keys back to the HA based on the NAI in User-Name attribute. Prime Access Registrar returns an Access-Reject if it does not find a valid session for the NAI during the user authentication and authorization or if there are other errors.

Prepaid and Hot-Lining

Prime Access Registrar supports prepaid and hot-lining flows for WiMAX. These are supported by the existing mechanisms.

Configuring WiMAX in Cisco Prime Access Registrar

A new service type named **wimax** will be used for the WiMAX feature in Prime Access Registrar. **aregcmd** command is used to configure WiMAX in Prime Access Registrar. WiMAX service contains—Session Manager (with a session-cache resource manager and HA resource manager), Query Service that is connected to the session manager configured for this service, and Prepaid Service, which are required to connect all the flows appearing in Prime Access Registrar for WiMAX. This service will be used as a container for the new key generation modules and the existing modules such as EAP services.

Configuring WiMAX in Prime Access Registrar involves configuration of:

- Resource Manager for WiMAX
- Session Manager for WiMAX
- Query Service for WiMAX
- WiMAX properties

This section contains the following topics:

- Configuring the Resource Manager for WiMAX
- Configuring the Session Manager for WiMAX
- Configuring the Query Service for WiMAX
- Configuring WiMAX
Configuring the Resource Manager for WiMAX

You must configure the following two Resource Managers:

- HA (home-agent or home-agent-ipv6)
- HA Cache (session-cache)

The HA Resource Manager must contain the IP ranges covering all the HA IP addresses that are to be assigned in round-robin. You must configure the HA Cache Resource Manager to cache the mobility keys (Table 10-3).

**Note**

The HA Resource Manager allocates the IP addresses to the HA. If you do not configure the HA Resource Manager properly, Prime Access Registrar will not generate some of the keys, which result in an Access-Reject by the NAS.

The following shows the sample configuration for HA:

```
[ /Radius/ResourceManagers/HA ]
Name = HA
Description =
Type = home-agent
Home-Agent-IPAddresses/
Entries 1 to 1 from 1 total entries
Current filter: <all>
209.165.200.225-209.165.200.254/
```

The following shows the sample configuration for HA Cache in HAAA:

```
[ /Radius/ResourceManagers/HA-Cache ]
Name = HA-Cache
Description =
Type = session-cache
OverwriteAttributes = TRUE
QueryKey = User-Name
PendingRemovalDelay = 10
AttributesToBeCached/
  1. WiMax-Session-ID
  2. hHA-RK-Key
  3. hHA-RK-SPI
  4. MN-hHA-MIP4-Key
  5. hHA-RK-Lifetime
  6. MIP-RK
```

The following shows the sample configuration for HA Cache in VAAA:

```
[ /Radius/ResourceManagers/HA-Cache ]
Name = HA-Cache
Description =
Type = session-cache
OverwriteAttributes = TRUE
QueryKey = User-Name
PendingRemovalDelay = 10
AttributesToBeCached/
  1. vHA-RK-Key
  2. vHA-RK-SPI
  3. MN-vHA-MIP4-Key
  4. vHA-RK-Lifetime
```

When the OverwriteAttributes value is set as TRUE, the newly generated mobility keys will be cached with the session record. By default, the value is FALSE.
The HA-RK-Lifetime attribute type must be of type STRING instead of UINT32 under /Radius//advanced/attribute\ dictionary/vendor-Specific/vendors/wimax/subAttribute\ Dictionary.

**Note**
For generating RRQ-MN-HA key, we must configure MIP-RK in the AttributesToBeCached list.

### Configuring the Session Manager for WiMAX

Before configuring WiMAX service, you must configure a session manager for WiMAX with a HA and session cache resource manager. The following shows an example configuration of a session manager with HA and session cache resource managers.

```
[ /Radius/SessionManagers/session-mgr-2 ]
  Name = session-mgr-2
  Description =
  IncomingScript =
  OutgoingScript =
  AllowAccountingStartToCreateSession = FALSE
  SessionTimeOut =
  PhantomSessionTimeOut =
  SessionKey =
  ResourceManagers/
    1. HA-Cache
    2. HA
```

**Note**
If a default session manager is configured with the same key as that of the WiMAX session manager, the incoming WiMAX request will fail.

### Configuring the Query Service for WiMAX

When you configure a query service for the WiMAX service in Prime Access Registrar, you must refer it to the WiMAX Session Manager that you created. While configuring WiMAX, you must refer the WiMAX-Query-Service parameter to a valid Query Service.

You must configure the Query key as the User-Name attribute, which contains the NAI. You must also configure the query service to return all the relevant mobility keys as described in Table 10-5.

#### Table 10-5  Mobility Keys

<table>
<thead>
<tr>
<th>Key</th>
<th>Generated By</th>
<th>Used At</th>
</tr>
</thead>
<tbody>
<tr>
<td>MN-HA-CMIP4</td>
<td>MN and HAAA</td>
<td>HA and MN</td>
</tr>
<tr>
<td>MN-HA-PMIP4</td>
<td>MN and HAAA</td>
<td>HA and PMIP4 client</td>
</tr>
<tr>
<td>MN-HA-CMIP6</td>
<td>MN and HAAA</td>
<td>MN and HA</td>
</tr>
<tr>
<td>FA-RK</td>
<td>MN and HAAA</td>
<td>MN and Authenticator</td>
</tr>
<tr>
<td>MN-FA</td>
<td>MN and Authenticator</td>
<td>FA and MN or PMIP4 client</td>
</tr>
<tr>
<td>HA-RK</td>
<td>HAAA or VAAA</td>
<td>HA and Authenticator</td>
</tr>
<tr>
<td>FA-HA</td>
<td>HA and Authenticator</td>
<td>HA and FA</td>
</tr>
</tbody>
</table>
The following shows a sample configuration for a WiMAX Query Service:

```plaintext
[../haQueryService ]
Name = haQueryService
Description =
Type = radius-query
IncomingScript~ =
OutgoingScript~ =
SessionManagersToBeQueried/
  1. session-mgr-2
AttributesToBeReturned/
  1. WiMax-Session-ID
  2. HA-RK-Key
```

**Note**

If AttributesToBeReturned is not configured, all the cached attributes will be returned.

### Configuring WiMAX

When you configure the WiMAX service under /Radius/Services, you must set its type to `wimax` and provide the following configuration options:

```plaintext
[ //localhost/Radius/Services/wimax ]
Name = WiMAX
Description =
Type = WiMAX
IncomingScript~ =
OutgoingScript~ =
OutagePolicy~ = RejectAll
OutageScript~ =
HA-RK-Key = cisco112
HA-RK-LifeTime = 60
WiMAX-Authentication-Service = None
WiMAX-Session-Manager = None
WiMAX-Query-Service = None
WiMAX-Prepaid-Service = None
Allow-HAAA-To-Include-Keys = TRUE
Require-MSK = False
```

The syntax to generate the a WiMAX request from radclient is

```plaintext
simple_wimax_asn_test bob(username) bob(password)
```

<table>
<thead>
<tr>
<th><strong>Table 10-6</strong> WiMAX Service Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Parameter</strong></td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Type</td>
</tr>
<tr>
<td>IncomingScript</td>
</tr>
<tr>
<td>OutgoingScript</td>
</tr>
<tr>
<td>OutagePolicy</td>
</tr>
</tbody>
</table>
WiMAX - OMA-DM Provisioning Support with BEK Key

In addition to WiMax subscriber authentication, the Prime Access Registrar generates and caches the Bootstrap Encryption Key (BEK) when it receives the authentication request from the unprovisioned WiMax subscriber/device. Prime Access Registrar can identify the unprovisioned device either by looking the special pattern in Access-Request or by performing explicit database lookup.

The BEK key derived from EMSK is calculated as follows:

### Table 10-6 WiMAX Service Parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OutageScript</td>
<td>Optional. If you set this property to the name of a script, Prime Access Registrar runs it when an outage occurs. This property allows you to create a script that notifies you when the RADIUS server detects a failure.</td>
</tr>
<tr>
<td>HA-RK-Key</td>
<td>Used as the base key to generate random HA-RK-Key for all the HAs that are configured in Prime Access Registrar. By default, the value is cisco112. You can change this value.</td>
</tr>
<tr>
<td>HA-RK-LifeTime</td>
<td>Used as time (in minutes) to regenerate the HA-RK-Keys based on its lifetime.</td>
</tr>
<tr>
<td>WiMAX-Authentication-Service</td>
<td>A valid EAP service which can be used for WiMAX authentication. By default, this value is none. For VAAA, it should be configured with valid radius proxy service.</td>
</tr>
<tr>
<td>WiMAX-Session-Manager</td>
<td>A valid session manager which has HA and HA Cache as resource managers. By default, this value is none.</td>
</tr>
<tr>
<td>WiMAX-Query-Service</td>
<td>A valid RADIUS query service configured with WiMAX session manager. By default, this value is none.</td>
</tr>
<tr>
<td>WiMAX-Prepaid-Service</td>
<td>A valid prepaid service can be given to carry out the prepaid functionality of WiMAX. Otherwise this value is set to none.</td>
</tr>
<tr>
<td>Allow-HAAA-To-Include-Keys</td>
<td>If this is set, the HAAA will include the hHA-RK-Key, hHA-RK-SPI and hHA-RK-Lifetime in the Access-Accept. Otherwise, those attributes will not be in the Access-Accept. By default this value is True.</td>
</tr>
<tr>
<td>Require-MSK</td>
<td>If this is set, the MSK will be provided by the AAA server as a result of successful EAP-Authentication. By default, this value is False.</td>
</tr>
</tbody>
</table>
BEK = the 16 most significant (leftmost) octets of HMAC-SHA256(EMSK, “bek@wimaxforum.org”).

When Prime Access Registrar receives the accounting start packet for the unprovisioned device,

1. IP, MAC address, and BEK of the unprovisioned device notifies the OMA-DM server to initiate the provisioning.

2. Prime Access Registrar maintains the IP address to MAC address association using web-service until it receives the provisioning complete message from the OMA-DM server.

The Backend Portal queries the Prime Access Registrar web-service for this unprovisioned device MAC address by giving the device IP address and also the OMA-DM server request the Prime Access Registrar web-service to validate the MAC to IP address association.

The communication between Prime Access Registrar and OMA-DM/Portal server is through web-service by using SOAP over HTTPS. It is assumed that the OMA-DM server (or a mediation function) will have a web-service using which AR can communicate.

**Configuring the WiMax-Provisioning**

To configure WiMax provisioning:

---

**Step 1**

Configure a script object, such as wimax-provision.

```
[ //localhost/Radius/Scripts/wimax-provision ]
Name = wimax-provision
Description = Language = rex
```

--> set FileName to ‘libProvisioning.so’

```
set FileName /cisco-ar/scripts/radius/rex/libProvisioning.so
```

--> set EntryPoint 'ProvisionedDeviceLookup'

```
set EntryPoint ProvisionedDeviceLookup
```

--> set InitEntryPoint 'InitializeProvisioning'

```
set InitEntryPoint InitializeProvisioning
```

--> set InitEntryPointArgs to 'ldap:wimax'

```
set InitEntryPointArgs ldap:wimax
```

The file libProvisioning.so is come up with Prime Access Registrar kit. You have to copy it into /cisco-ar/scripts/radius/rex path. Entry-point ProvisionedDeviceLookup literally looks up a datastore to check if the user is provisioned. InitEntryPoint 'InitializeProvisioning' takes care of all initialization work for entry point. InitEntryPointArgs 'ldap:wimax' says the user look up to be performed against ldap datastore. Oracle datastore can also be used wherein you have to give this property to 'oracle:wimax'.
Step 2 Configure the configured script object to the server's incoming scripting point.

```
set IncomingScript wimax-provision
```

**Is**

```
[ //localhost/Radius ]
Name = Radius
Description =
Version = 5.0.0
IncomingScript~ = provision
OutgoingScript~ =
```

Step 3 Webclient setup

Create a script object which calls the Prime Access Registrar's wimax-provisioning webservice.

```
[ //localhost/Radius/Scripts/WebServicecall ]
Name = WebServicecall
Description =
Language = rex
Filename = libProvisioning.so
EntryPoint = WebServiceCall
InitEntryPoint =
InitEntryPointArgs =
```

Entry point should be set to WebServiceCall.

Step 4 Save the configuration:

```
save
```

Step 5 Reload the configuration:

```
reload
```
Using Extension Points

This chapter describes how to use Cisco Prime Access Registrar (Prime Access Registrar) scripting to customize your RADIUS server. At specific points during a Prime Access Registrar Request-Response packet flow, service providers can initiate scripts using REX (C/C++), Java, TCL, or internal script interfaces, to customize or modify the packet flow. When the script finishes, the packet flow continues with the next step. You can write scripts to affect the way Prime Access Registrar handles and responds to requests and to change the behavior of Prime Access Registrar after a script is run.

All scripts reference the three dictionaries listed below, which are data structures that contain key/value pairs.

- **Request dictionary**—contains all of the attributes from the access-request or other incoming packets, such as the username, password, and service hints.
- **Response dictionary**—contains all of the attributes in the access-accept or other response packets. As these are the attributes the server sends back to the NAS, you can use this dictionary to add or remove attributes.
- **Environment dictionary**—contains well-known keys whose values enable scripts to communicate with Prime Access Registrar or to communicate with other scripts.

This chapter contains the following sections:

- Determining the Goal of the Script, page 11-1
- Writing the Script, page 11-2
- Adding the Script Definition, page 11-5
- About the Tcl/Tk 8.3 Engine, page 11-6
- Cisco Prime Access Registrar Scripts, page 11-6
- Extension Points in Cisco Prime Access Registrar, page 11-16

**Note**
Cisco is not liable for scripts developed by clients. See Client Scripting, page 1-7 in “Overview”.

**Determining the Goal of the Script**

The goal of the script and its scripting point are tied together. For example, when you want to create a script that performs some special processing of a username before it is processed by the Prime Access Registrar server, you would reference this script as an incoming script.
When on the other hand, you would like to affect the response, such as setting a specific timeout when there is not one, you would reference the script as an *outgoing* script.

In order to be able to create a script, you need to understand the way Prime Access Registrar processes client requests. Prime Access Registrar processes requests and responses in a hierarchical fashion; incoming requests are processed from the most general to the most specific levels, whereas, outgoing responses are processed from the most specific to the most general levels. Extension points are available at each level.

An incoming script can be referenced at each of the following extension points:

- RADIUS server
- Vendor (of the immediate client)
- Client (individual NAS)
- NAS-Vendor-Behind-the-Proxy
- Client-Behind-the-Proxy
- Remote Server (of type RADIUS)
- Service

An authentication or authorization script can be referenced at each of the following extension points:

- Group Authentication
- User Authentication
- Group Authorization
- User Authorization

The outgoing script can be referenced at each of the following extension points:

- Service
- Client-Behind-the-Proxy
- NAS-Vendor-Behind-the-Proxy
- Client (individual NAS)
- NAS Vendor
- RADIUS server

**Writing the Script**

You can write scripts in Tcl, REX, Java, or as shared libraries using C or C++. In this section, the scripts are shown in Tcl.

**Writing the Script**

To write a script:

1. Create the Tcl source file using an editor.
2. Give it a name.
3. Define one or more procedures, using the following syntax:

   ```tcl
   proc name {request response environment}
   {Body}
   ```
Step 4 Create the body of the script.

Step 5 Save the file and copy it to the /opt/CSCOar/scripts/radius/tcl directory, or to the location you chose when you installed Prime Access Registrar.

Note You can also use the Prime Access Registrar GUI or CLI to write internal scripts, by which you can add, modify, or delete attribute in the request, response, and environment dictionaries for RADIUS, Diameter, and TACACS+. For more information, see Internal Scripts, page 11-14.

Choosing the Type of Script

When you create a script, you can use any one or all of the three dictionaries: Request, Response, or Environment. Here is what each dictionary does it for you,

- When you use the Request dictionary, you can modify the contents of a NAS request. Scripts that use the Request dictionary are usually employed as incoming scripts.
- When you use the Response dictionary, you can modify what the server sends back to the NAS. These scripts are consequently employed as outgoing scripts.
- When you use the Environment dictionary, you can do the following:
  - Affect the behavior of the server after the script is run. For example, you can use the Environment dictionary to decide which of the multiple services to use for authorization, authentication, and accounting.
  - Communicate among scripts, as the scripts all share these three dictionaries. For example, when a script changes a value in the Environment dictionary, the updated value can be used in a subsequent script.

The following sections show scripts examples of all the three dictionaries:

- Request Dictionary Script
- Response Dictionary Script
- Environment Dictionary Script

Request Dictionary Script

The Request Dictionary script is referenced from the server’s IncomingScript scripting point. It checks to see whether the request contains a NAS-Identifier or a NAS-IP-Address. When it does not, it sets the NAS-IP-Address from the request’s source IP address.

```
proc MapSourceIPAddress {request response environment} {
    if { ! ( [ $request containsKey NAS-Identifier ] ||
        [ $request containsKey NAS-IP-Address ] ) } {
        $request put NAS-IP-Address [ $environment get Source-IP-Address ]
    }
}
```

Tcl scripts interpret $request arguments as active commands that can interpret strings from the Request dictionary, which contains keys and values.
The `containsKey` method has the syntax: `<$dict> containsKey <attribute>`. In this example, `<$dict>` refers to the Request dictionary and the attributes `NAS-identifier` and `NAS-IP-Address`. The `containsKey` method returns 1 when the dictionary contains the attribute, and 0 when it does not. Using the `containsKey` method prevents you from overwriting an existing value.

The `put` method has the syntax: `<$dict> put <attribute value>[<index>]`. In this example, `<$request>` refers to the Request dictionary and the attribute is `NAS-IP-Address`. The `put` method sets the NAS’s IP address attribute.

The `get` method has the syntax: `<$dict> get <attribute>`. In this example, `<$dict>` refers to the Environment dictionary and `<attribute>` is the `Source-IP-Address`. The `get` method returns the value of the attribute from the environment dictionary.

For a list of the methods you can use with scripts, see Appendix A, “Cisco Prime Access Registrar Tcl, REX, and Java Dictionaries.” They include `get`, `put`, and others.

**Response Dictionary Script**

This script is referenced from either the user record for users whose sessions are always PPP, or from the script, `AuthorizeService`, which checks the request to determine which service is desired. The script merges the Profile named `default-PPP-users` into the Response dictionary.

```tcl
proc AuthorizePPP {request response environment} {  
    $response addProfile default-PPP-users
}
```

The `addProfile` method has the syntax: `<$dict> addProfile `<profile>`[<mode>]`. In this example, `<$dict>` refers to the Response dictionary and the profile is `default-PPP-users`. The script copies all of the attributes of the Profile `<profile>` into the dictionary.

**Environment Dictionary Script**

This script is referenced from the NAS Incoming Script scripting point. It looks for a realm name on the username attribute to determine which AAA services should be used for the request. When it finds `@radius`, it selects a set of AAA services that will proxy the request to a remote RADIUS server. When it finds `@tacacs`, it selects the Authentication Service that will proxy the request to a TACACS server for authentication. For all of the remaining usernames, it uses the default Service (as specified in the configuration by the administrator).

Note the function, `regsub`, is a Tcl function.

```tcl
proc ParseProxyHints {request response environment} {  
    set userName [ $request get User-Name ]
    if { [ regsub "@radius" $userName "" newUserName ] } {  
        $request put User-Name $newUserName
        $radius put Authentication-Service "radius-proxy"
        $radius put Authorization-Service "radius-proxy"
        $radius put Accounting-Service "radius-proxy"
    } else {  
        if { [ regsub "@tacacs" $userName "" newUserName ] } {  
            $request put User-Name
            $radius put Authentication-Service "tacacs-client"
        }
    }
}
```
Adding the Script Definition

After you have written the script, you must add the script definition to the Prime Access Registrar’s script Configuration directory so it can be referenced. Adding the script definition involves:

- Specifying the script definition; it must include the following:
  - **Name**—used in other places in the configuration to refer to the script. It must be unique among all other scripts.
  - **Language**—can be either Tcl or REX (shared libraries)
  - **Filename**—the name you used when you created the file
  - **EntryPoint**—the function name.

The **Name** and the **EntryPoint** can be the same name, however they do not have to be.

- Choosing a scripting point; nine exist for incoming and outgoing scripts. These include:
  - the server
  - the vendor of the immediate client
  - the immediate client
  - the vendor of the specific NAS
  - the specific NAS
  - the service (rex or tcl)
  - the group (only AA scripts)
  - the user record (only AA scripts)
  - remote server (only type RADIUS)

The rule of thumb to use in determining where to add the script is the most general scripts should be on the outermost points, whereas the most specific scripts should be on the innermost points.

**Note**

The client and the NAS are the same entity, unless the immediate client is acting as a proxy for the actual NAS.

This section contains the following topics:

- Adding the Example Script Definition
- Choosing the Scripting Point
- Testing the Script

Adding the Example Script Definition

In the server configuration a Scripts directory exists. You must add the script you created to this directory. To add the ParseProxyHints script to the Prime Access Registrar server, enter the following command and supply the following information:

```
Name=ParseProxyHints
Description=ParseProxyHints
Language=tcl
```
**Choosing the Scripting Point**

As the example script, `ParseProxyHints`, applies to a specific NAS (NAS1), the entry point should be that NAS. To specify the script at this scripting point, enter:

```
aregcmd set /Radius/Clients/NAS1/IncomingScript ParseProxyHints
```

**Testing the Script**

To test the script, you can use the `radclient` command, which lets you create and send packets. For more information about the `radclient` command, see Chapter 2, “Using the aregcmd Commands.”

**About the Tcl/Tk 8.3 Engine**

Prime Access Registrar uses Tcl engine is version Tcl/Tk8.3. Since the Tcl/Tk8.3 engine supports a multi-threading application environment, it can achieve much better performance than Tcl/Tk7.6. Tcl/Tk8.3 also performs *byte-compile*, so no runtime interpretation is required.

**Cisco Prime Access Registrar Scripts**

The Prime Access Registrar scripts are stored in `/localhost/Radius/Scripts`. Most of the scripts are written in the RADIUS Extension language (REX). Some scripts are provided in both REX and Tcl. The scripts written in Tcl all begin with the letter `t` followed by their functional name. The Tcl scripts are listed below:

- tACMEOutgoingScript
- tAuthorizePPP
- tAuthorizeService
- tAuthorizeTelnet
- tMapSourceIPAddress
- tParseAARealm
- tParseAASRealm
- tParseProxyHints
- tParseServiceAndAARealmHints
- tParseServiceAndAAASRealmHints
- tParseServiceAndAARealmHints
- tParseServiceAndProxyHints
- tParseServiceHints
You can use the Prime Access Registrar GUI to write internal scripts, by which you can add, modify, or delete attribute in the request, response, and environment dictionaries for RADIUS, Diameter, and TACACS+. For more information about configuring internal scripts by using the GUI, see Scripts, page 3-18.

This section contains the following topics:

- ACMEOutgoingScript
- AltigaIncomingScript
- AltigaOutgoingScript
- ANAAAOutgoing
- AscendIncomingScript
- AscendOutgoingScript
- AuthorizePPP
- AuthorizeService
- AuthorizeSLIP
- AuthorizeTelnet
- CabletronIncoming
- CabletronOutgoing
- CiscoIncoming
- CiscoOutgoing
- CiscoWithODAPIncomingScript
- ExecCLIDRule
- ExecDNISRule
- ExecFilterRule
- ExecNASIPRule
- ExecRealmRule
- ExecTimeRule
- LDAPOutage
- MapSourceIPAddress
- ParseAAARealm
- ParseAAASRealm
- ParseAAARealm
- ParseAASRealm
- ParseProxyHints
- ParseServiceAndAAARealmHints
- ParseServiceAndAAASRealmHints
- ParseServiceAndAAARealmHints
- ParseServiceAndAASRealmHints
- ParseServiceAndProxyHints
Cisco Prime Access Registrar Scripts

- ParseServiceHints
- ParseTranslationGroupsByCLID
- ParseTranslationGroupsByDNIS
- ParseTranslationGroupsByRealm
- UseCLIDAsSessionKey
- USRIncomingScript
- USRIncomingScript-IgnoreAccountingSignature
- USROutgoingScript
- Internal Scripts, page 11-14

ACMEOutgoingScript

ACMEOutgoingScript is referenced from Vendor ACME for the outgoing script. If the Prime Access Registrar server accepts this Access-Request and the response does not yet contain a Session-TimeOut, set it to 3600 seconds.

AltigaIncomingScript

AltigaIncomingScript maps Altiga-proprietary attributes to Prime Access Registrar’s global attribute space.

AltigaOutgoingScript

AltigaOutgoingScript maps Altiga attributes from Prime Access Registrar’s global attribute space to the appropriate Altiga-proprietary attributes.

ANAAAOutgoing

ANAAAOutgoing can be referenced from either the client or vendor outgoing scripting point to be used in HRPD/EV-DO networks where Prime Access Registrar is the Access Network (AN) AAA server. ANAAAOutgoing checks to see if the response contains the Callback-Id attribute. If the response contains the Callback-Id attribute and the value is less than 253 characters, ANAAAOutgoing prefixes a zero (0) to the value. For example, it changes "112" into "0112." The ANAAAOutgoing script always returns REX_OK.

AscendIncomingScript

AscendIncomingScript maps Ascend-proprietary attributes to Prime Access Registrar’s global attribute space.
AscendOutgoingScript

- AscendOutgoingScript maps Ascend attributes from Prime Access Registrar’s global attribute space to the appropriate Ascend-proprietary attributes.

AuthorizePPP

- AuthorizePPP is referenced from either the user record for users who’s sessions are always PPP or from the script AuthorizeService, which checks the request to determine which service is desired. This script merges in the Profile named “default-PPP-users” into the response dictionary.

AuthorizeService

- AuthorizeService is referenced from user record for users who’s sessions might be PPP, SLIP or Telnet depending on how they are connecting to the NAS. This script checks the request to determine which service is desired. If it is telnet, it calls the script AuthorizeTelnet. If it is PPP, it calls the script AuthorizePPP. If it is SLIP, it calls the script AuthorizeSLIP. If it is none of these, it rejects the request.

AuthorizeSLIP

- AuthorizeSLIP is referenced from either the user record for users who’s sessions are always SLIP or from the script AuthorizeService, which checks the request to determine which service is desired. This script merges in the Profile named “default-SLIP-users” into the response dictionary.

AuthorizeTelnet

- AuthorizeTelnet is referenced from either the user record for users who’s sessions are always telnet or from the script AuthorizeService, which checks the request to determine which service is desired. This script merges in the Profile named “default-Telnet-users” into the response dictionary.

CabletronIncoming

- CabletronIncoming maps Cabletron-proprietary attributes to Prime Access Registrar’s global attribute space.

CabletronOutgoing

- Use CabletronOutgoing to map Cisco-proprietary attributes from Prime Access Registrar’s global attribute space to the appropriate Cabletron-proprietary attributes.

CiscoIncoming

- Use CiscoIncoming to map Cisco-proprietary attributes to Prime Access Registrar’s global attribute space.
CiscoOutgoing

Use CiscoOutgoing to map Cisco-proprietary attributes from Prime Access Registrar’s global attribute space to the appropriate Cabletron-proprietary attributes.

CiscoWithODAPIncomingScript

Use CiscoWithODAPIncomingScript to map Cisco-proprietary attributes to Prime Access Registrar’s global attribute space and to map ODAP requests to the appropriate services and session managers. CiscoWithODAPIncomingScript checks the incoming NAS-Identifier sent by the client. If the NAS-Identifier does not equal odap-dhcp, the request is not an ODAP request. If the request is not an ODAP request, the script does no more ODAP-specific processing, and calls CiscoIncomingScript to allow it to process the request.

If the request is an ODAP request, CiscoWithODAPIncomingScript removes the NAS-Identifier attribute because it is no longer required. The script then sets the Authentication-Service and the Authorization-Service to odap-users and sets the Accounting-Service to odap-accounting.

ExecCLIDRule

ExecCLIDRule is referenced from the policy engine to determine the authentication and authorization service and policy based on the CLID set in the policy engine.

ExecDNISRuler

ExecDNISRuler is referenced from the policy engine to determine the authentication and authorization service and policy based on the DNIS set in the policy engine.

ExecFilterRule

ExecFilterRule is referenced from the policy engine to determine whether a user packet should be rejected or not based on whether a special character like "#", "/", "\" or "?" shows up in the packet.

ExecNASIPRule

ExecNASIPRule is referenced from the policy engine to enable configuration of policies based on the incoming NAS-IP-Address. You can configure two attributes, client-ip-address and subnetmask, to match the incoming NAS-IP-Address and its subnet mask. If the attributes match, ExecNASIPRule sets the environment variables (if they are configured in that rule).

ExecRealmRule

ExecRealmRule is referenced from the policy engine to determine the authentication and authorization service and policy based on the realm set in the policy engine.


**ExecTimeRule**

ExecTimeRule either rejects or accepts Access Request packets based on the time range specified in a user's login profile. You can configure the TimeRange and AcceptedProfile attributes.

The format for the TimeRange is to set the allowable days followed by the allowable times, as in:

\[
\text{TimeRange} = \text{dateRange, timeRange}
\]

The dateRange can be in the form of a date, a range of allowable dates, a day, or a range of allowable days. The timeRange should be in the form of hh:mm-hh:mm.

Here are a few examples:

- `mon-fri,09:00-17:00`
  - Allows access Monday through Friday from 9 AM until 5 PM.

- `mon,09:00-17:00;tue-sat,12:00-13:00`
  - Allows access on Monday from 9 AM until 5 PM and from 12 noon until 1 PM on Tuesday through Saturday

- `mon,09:00-24:00;tue,00:00-06:00`
  - Allows access on Monday from 9 AM until Tuesday at 6 AM

- `1-13,10-17:00; 15,00:00-24:00`
  - Allows access from the first of the month until the thirteenth of the month from 10 AM until 5 PM and all day on the fifteenth of the month

**LDAPOutage**

LDAPOutage is referenced from LDAP Services as OutageScript. LDAPOutage logs when the LDAP binding is lost.

**MapSourceIPAddress**

MapSourceIPAddress is referenced from the Prime Access Registrar server's IncomingScript scripting point. MapSourceIPAddress checks to see if the request contains either a NAS-Identifier or a NAS-IP-Address. If not, this script sets the NAS-IP-Address from the request's source IP address.

The Tcl version of this script is tMapSourceIPAddress.

**ParseAAARealm**

ParseAAARealm is referenced from the NAS IncomingScript scripting point. It looks for a realm name on the username attribute as a hint of which AAA service should be used for this request. If @<realm> is found, the AAA service is selected which has the same name as the realm.
ParseAAASRealm

ParseAAASRealm is referenced from the NAS incoming script extension point. ParseAAASRealm looks for a realm name on the username attribute as a hint of which AAA service and which SessionManager should be used for this request. If @<realm> is found, the AAA service and SessionManager which have the same name as the realm are selected.

ParseAAARealm

ParseAAARealm is referenced from the NAS IncomingScript scripting point. It looks for a realm name on the username attribute as a hint of which authentication and authorization service should be used for this request. If @<realm> is found, it selects the AA service that has the same name as the realm and the DefaultAccountingService (as specified in the configuration by the administrator).

The Tcl version of this script is named tParseAAARealm.

ParseAASRealm

ParseAASRealm is referenced from the NAS IncomingScript scripting point. It looks for a realm name on the username attribute as a hint of which AA service and which SessionManager should be used for this request. If @<realm> is found, the AA service and the SessionManager which have the same name as the realm are selected. The Accounting service will be the DefaultAccountingService (as specified in the configuration by the administrator).

The Tcl version of this script is named tParseAASRealm.

ParseProxyHints

ParseProxyHints is referenced from the NAS IncomingScript scripting point. It looks for a realm name on the username attribute as a hint of which AAA services should be used for this request. If @radius is found, a set of AAA services is selected which will proxy the request to a remote radius server. If @tacacs is found, the AuthenticationService is selected that will proxy the request to a tacacs server for authentication. For any services not selected, the default service (as specified in the configuration by the administrator) will be used.

The Tcl version of this script is named tParseProxyHints.

ParseServiceAndAAARealmHints

ParseServiceAndAAARealmHints is referenced from the NAS IncomingScript scripting point. It calls both ParseServiceHints and ParseAAARealm.

The Tcl version of this script is named tParseServiceAndAAARealmHints.

ParseServiceAndAAASRealmHints

ParseServiceAndAAASRealmHints is referenced from the NAS IncomingScript scripting point. It calls both ParseServiceHints and ParseAAASRealm.
The Tcl version of this script is named tParseServiceAndAAASRealmHints.

**ParseServiceAndAAARealmHints**

ParseServiceAndAAARealmHints is referenced from the NAS IncomingScript scripting point. It calls both ParseServiceHints and ParseAARealm.
The Tcl version of this script is named tParseServiceAndAAARealmHints.

**ParseServiceAndAASRealmHints**

ParseServiceAndAASRealmHints is referenced from the NAS IncomingScript scripting point. It calls both ParseServiceHints and ParseAASRealm.
The Tcl version of this script is named tParseServiceAndAASRealmHints.

**ParseServiceAndProxyHints**

ParseServiceAndProxyHints is referenced from the NAS IncomingScript scripting point. It calls both ParseServiceHints and ParseProxyHints.
The Tcl version of this script is named tParseServiceAndProxyHints.

**ParseServiceHints**

ParseServiceHints is referenced from the NAS IncomingScript scripting point. Check to see if we are given a hint of the service type or the realm. If so, set the appropriate attributes in the request or radius dictionary to record the hint and rewrite the username to remove the hint.
The Tcl version of this script is named tParseServiceHints.

**ParseTranslationGroupsByCLID**

ParseTranslationGroupsByCLID is referenced from the policy engine to determine the incoming and outgoing translation groups based on CLID set in the policy engine so that the attributes can be added and/or filtered out by the configuration data set in MCD.

**ParseTranslationGroupsByDNIS**

ParseTranslationGroupsByDNIS is referenced from the policy engine to determine the incoming and outgoing translation groups based on realm set in the policy engine so that the attributes can be added/filtered out by the configuration data set in MCD.
ParseTranslationGroupsByRealm

ParseTranslationGroupsByRealm is referenced from the policy engine to determine the incoming and outgoing translation groups based on the realm set in the policy engine. ParseTranslationGroupsByRealm allows the attributes to be added or filtered out by the configuration data set in MCD.

UseCLIDAsSessionKey

UseCLIDAsSessionKey is used to specify that the Calling-Station-Id attribute should be used as the session key to correlate requests for the same session. This is a typical case for 3G mobile user session correlation.

USRIncomingScript

USRIncomingScript maps USR-proprietary attributes to Prime Access Registrar’s global attribute space.

USRIncomingScript-IgnoreAccountingSignature

USRIncomingScript-IgnoreAccountingSignature maps USR-proprietary attributes to Prime Access Registrar’s global attribute space and sets a flag to ignore the signature on Accounting-Request packets. Earlier versions of the USR RADIUS client did not correctly sign Accounting-Request packets.

USROutgoingScript

USROutgoingScript maps USR attributes from Prime Access Registrar’s global attribute space to the appropriate USR-proprietary attributes.

Internal Scripts

Prime Access Registrar allows you to write internal scripts, by which you can add, modify, or delete attributes in the request, response, and environment dictionaries for RADIUS, Diameter, and TACACS+. You can use the Prime Access Registrar GUI or CLI to configure the internal scripts.

Prime Access Registrar allows you to create script statements for the following operations:

- Simple Attribute Operation—allows you to add, modify, or delete an attribute value to the request, response, or environment dictionary
- Copy an Attribute—allows you to copy an attribute value from one dictionary to another
- Concatenate Operation—allows you to concatenate an attribute value from one dictionary to another
- Replace Operation—allows you to replace an attribute value from one dictionary to another
- Value Based Manipulations—allows you to manipulate attribute values in a dictionary based on a given text
- Log or Trace Messages—allows you to create different levels of log or trace messages
- I can do it myself—allows you to create your own script for the selected protocol

You can also use internal scripts as part of the FastRules workflow. For more information about FastRules, see Chapter 19, “Using FastRules to Process Packet Flow.”

To configure internal scripts using the Prime Access Registrar GUI, see Scripts, page 3-18.

**CLI to Configure Internal Scripts**

A sample CLI to configure internal script statements is given below:

```bash
--cd /r/scripts/test

//localhost/Radius/Scripts/test
Name = test
Description =
Language = internal
Statements/

--cd statements/

//localhost/Radius/Scripts/test/Statements
1. #req:User-Name=~(.*)(@[a-z]+.[a-z]+)~

//localhost/Radius/Scripts/test1
Name = test1
Description =
Language = internal
Statements/

--cd statements/

//localhost/Radius/Scripts/test1/Statements
1. -rsp:Framed-IP-Address=1.1.1.1

-->

//localhost/Radius/Scripts/test3/Statements
1. -rsp:Tacacs-AVpair=cmd=show running-config
2. -rsp:Tacacs-AVPair=aaa
3. +rsp:Tacacs-AVPair=aaa

-->

//localhost/Radius/Scripts/test4/Statements
1. -req:Cisco-AVPair=bbb

--cd ../..//test7/statements/

//localhost/Radius/Scripts/test7/Statements
1. #rsp:Framed-IPX-Network=2
2. +rsp:State=Delivered
3. -rsp:State
4. +req:Cisco-AVPair=aaa
5. #req:Cisco-AVPair=5
6. #rsp:Framed-IPX-Network=req:Cisco-AVPair
7. -req:Cisco-AVPair
```
Extension Points in Cisco Prime Access Registrar

Table 11-1 lists the scripting points used in Prime Access Registrar. You can find the scripting point information in the file /cisco-ar/examples/rexscript/rex.h.

Note the following:
- All the scripting points can be used for processing packets of type RADIUS, Diameter, or TACACS+.
- The packet type—RADIUS, Diameter, or TACACS+—can be obtained from the ‘Request-Type’ or ‘Response-Type’ attribute of the environment dictionary. For more details, see Appendix B, “Environment Dictionary.”

<table>
<thead>
<tr>
<th>Service</th>
<th>Scripting Point Number</th>
<th>Scripting Point</th>
<th>Description</th>
<th>Sample Use Case (if applicable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td>1</td>
<td>Server incoming</td>
<td>Script runs for every request packet.</td>
<td>To inspect each packet and determine the kind of service (authentication, authorization, or accounting) to be provided.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Command-line interface (CLI) configuration path: /Radius/IncomingScript</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Remote Server incoming</td>
<td>Script runs when a packet is received from a remote server.</td>
<td>To add/modify/delete attributes coming from the remote server response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: /Radius/RemoteServers/&lt;remote server name&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>—</td>
<td>2</td>
<td>Vendor incoming</td>
<td>Script runs only for requests from all the clients mapped to a particular vendor.</td>
<td>To translate vendor proprietary attributes to RADIUS/Diameter vendor-specific attributes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: /Radius/Vendors/&lt;vendorname&gt;/IncomingScript</td>
<td></td>
<td></td>
</tr>
<tr>
<td>—</td>
<td>3</td>
<td>Client incoming</td>
<td>Script runs only for requests from the specified client (router, NAS, and so on).</td>
<td>To choose AAA service based on an NAS IP address.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: /Radius/Clients/&lt;clientname&gt;/IncomingScript</td>
<td></td>
<td></td>
</tr>
<tr>
<td>—</td>
<td>5</td>
<td>Service incoming</td>
<td>Script runs when a service is called.</td>
<td>To add/modify/delete attributes before calling a particular service.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: /Radius/Services/&lt;servicename&gt;/IncomingScript</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 11-1  Scripting Points in Prime Access Registrar (continued)

<table>
<thead>
<tr>
<th>Service</th>
<th>Scripting Point Number</th>
<th>Scripting Point</th>
<th>Description</th>
<th>Sample Use Case (if applicable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local</td>
<td>6</td>
<td>User group authentication</td>
<td>Script runs for authentication requests for a user belonging to a particular user group.</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: <code>/Radius/UserGroups/&lt;name&gt;/AuthenticationScript</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>User record authentication</td>
<td>Script runs for authentication requests for a particular user.</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: <code>/Radius/UserLists/&lt;Userlistname&gt;/&lt;username&gt;/AuthenticationScript</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>User group authorization</td>
<td>Script runs for authorization requests for a user belonging to a particular user group.</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: <code>/Radius/UserGroups/&lt;name&gt;/AuthorizationScript</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>User record authorization</td>
<td>Script runs for authorization requests for a particular user.</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: <code>/Radius/UserLists/&lt;Userlistname&gt;/&lt;username&gt;/AuthorizationScript</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>Service outgoing</td>
<td>Script runs after a service is executed.</td>
<td>To add/modify/delete attributes before sending the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: <code>/Radius/Services/&lt;serviceName&gt;/OutgoingScript</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>Client outgoing</td>
<td>Script runs when sending a response against the request received from a specific client (router, NAS, and so on.)</td>
<td>To add/modify/delete attributes before sending a response to the particular client.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: <code>/Radius/Clients/&lt;clientname&gt;/OutgoingScript</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>Vendor outgoing</td>
<td>Script runs when sending a response against the requests received from all the clients mapped to a particular vendor.</td>
<td>To translate RADIUS/Diameter vendor-specific attributes to the appropriate vendor proprietary attributes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: <code>/Radius/Vendors/&lt;vendorname&gt;/OutgoingScript</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>Server outgoing</td>
<td>Script runs for every response packet.</td>
<td>To add/modify/delete attributes before sending the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: <code>/Radius/OutgoingScript</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Remote Server outgoing</td>
<td>Script runs when a packet is sent out to a remote server.</td>
<td>To add/modify/delete attributes in a packet going to a remote server.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path: <code>/Radius/RemoteServers/&lt;remote server name&gt;</code></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 11-1 Scripting Points in Prime Access Registrar (continued)

<table>
<thead>
<tr>
<th>Service</th>
<th>Scripting Point Number</th>
<th>Scripting Point Description</th>
<th>Sample Use Case (if applicable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remote</td>
<td>16</td>
<td>Remote server outage</td>
<td>To add/modify/delete attributes when the remote servers defined in the service are down.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/Radius/Services/&lt;service name&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Script runs when the remote server is down.</td>
<td></td>
</tr>
<tr>
<td>Session Manager</td>
<td>19</td>
<td>Session manager incoming</td>
<td>To add attributes to the cache for a user before the session manager is called.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/Radius/SessionManagers/&lt;session manager name&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Script runs to direct requests to be processed by a specific session manager after authentication and authorization.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>Session manager outgoing</td>
<td>To translate the session cache attribute before sending the response.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/Radius/SessionManagers/&lt;session manager name&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Script runs to direct responses from a specific session manager after authentication and authorization.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>21</td>
<td>Virtual server outgoing</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>—</td>
<td></td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>Global title translation</td>
<td>To define GTT based on incoming International Mobile Subscriber Identity (IMSI) ranges.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/Radius/RemoteServers/&lt;remote server name&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Script runs before sending out request to Home Registration Register (HLR) to perform Global Title Translation (GTT).</td>
<td></td>
</tr>
<tr>
<td>SIGTRAN/ M3UA</td>
<td>23</td>
<td>Quintet generation</td>
<td>To fetch quintet information from a configured file based on the incoming IMSI.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/Radius/Services/&lt;EAP-AKA service name&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Script runs to generate quintets by using a simulator for EAP-AKA service.</td>
<td></td>
</tr>
<tr>
<td>EAP-AKA</td>
<td>24</td>
<td>IMSI translation</td>
<td>To modify the incoming IMSI ranges before sending a request to a Signal Transfer Point (STP)/HLR.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/Radius/RemoteServers/&lt;remote server name&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Script runs to change the incoming IMSI.</td>
<td></td>
</tr>
<tr>
<td>Translation</td>
<td>25</td>
<td>FastRule</td>
<td>To add an attribute in the same flow after authentication and authorization of a packet.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Script is configured within a fast rule execution path.</td>
<td></td>
</tr>
</tbody>
</table>
### Table 11-1  Scripting Points in Prime Access Registrar (continued)

<table>
<thead>
<tr>
<th>Service</th>
<th>Scripting Point Number</th>
<th>Scripting Point</th>
<th>Description</th>
<th>Sample Use Case (if applicable)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request Translation</td>
<td>26</td>
<td>Prerequest translation</td>
<td>To add/modify/delete incoming RADIUS/Diameter attribute values in the request before translation.</td>
<td>To add/modify/delete incoming RADIUS/Diameter attribute values in the request before translation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path:</td>
<td>/Radius/Services/&lt;service name&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path:</td>
<td>/Radius/Services/&lt;service name&gt;</td>
<td></td>
</tr>
<tr>
<td>Response Translation</td>
<td>28</td>
<td>Preresponse translation</td>
<td>To add/modify/delete Diameter/RADIUS attribute values in the response before translation.</td>
<td>To add/modify/delete Diameter/RADIUS attribute values in the response before translation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI configuration path:</td>
<td>/Radius/Services/&lt;service name&gt;</td>
<td></td>
</tr>
</tbody>
</table>
Using Replication

This chapter provides information about how to use the replication feature in Cisco Prime Access Registrar (Prime Access Registrar).

This chapter contains the following sections:

- Replication Overview
- How Replication Works
- Replication Configuration Settings
- Setting Up Replication
- Replication Example
- Full Resynchronization
- Replication Setup with More Than One Slave
- Frequently Asked Questions
- Replication Log Messages

**Note**

When using replication, use the `aregcmd` command-line interface to make configuration changes to the Prime Access Registrar server. Replication is not supported when using the GUI.

**Replication Overview**

Prime Access Registrar replication feature can maintain identical configurations on multiple machines simultaneously. When replication is properly configured, changes an administrator makes on the primary or *master* machine are propagated by Prime Access Registrar to a secondary or *slave* machine.

Replication eliminates the need to have administrators with multiple Prime Access Registrar installations make the same configuration changes at each of their installations. Instead, only the master's configuration need be changed and the slave is automatically configured eliminating the need to make repetitive, error-prone configuration changes for each individual installation. In addition to enhancing server configuration management, using replication eliminates the need for a hot-standby machine.

Using a hot-standby machine is a common practice to provide more fault-tolerance where a fully-installed and configured system stands ready to takeover should the primary machine fail. However, a system setup for hot-standby is essentially an idle machine only used when the primary
system fails. Hot-standby or secondary servers are expensive resources. Employing Prime Access Registrar’s replication feature, both servers can perform RADIUS request processing simultaneously, eliminating wasted resources.

The replication feature focuses on configuration maintenance only, not session information or installation-specific information such as Administrator, Interface, Replication or Advanced machine-specific configuration changes. These configuration items are not replicated because they are specific to each installation and are not likely to be identical between master and slave. While changes to Session Managers, Resource Manager, and Remote Servers are replicated to the slave and stored in the slave’s configuration database, they are not hot-configured on the slave (see Hot Configuration Detailed below for more information).

Changes should be made only on the master server. Making changes on a slave server will not be replicated and might result in an unstable configuration on the slave. Any changes made using replication will not be reflected in existing aregcmd sessions. aregcmd only loads its configuration at start up; it is not dynamically updated. For example, if aregcmd is running on the slave, and on the master aregcmd is used to add a client, the new client, while correctly replicated and hot-configured, will not be visible in the slave's aregcmd until aregcmd is exited and restarted.

When there is a configuration change, the master server propagates the change set to all member servers over the network. All member servers have to update their configuration after receiving the change set notifications from master server. Propagating the change set to a member server involves multiple packet transfer from the master server to the member because the master serve has to convey all the configuration changes to the member. The number of packets to be transferred depends on the size of the change set.

After receiving a change set notification, the member server will go offline before applying the change set received from master server. This state is indicated by the log message Radius Server is offline in name_radius_1_log file. When the change set is successfully applied, the member server goes up automatically. This is indicated by the log message Radius Server is online in name_radius_1_log file. When the member server goes offline to apply the change set, no incoming packets are processed.

Due to the number of packets to be transferred in the change set and the amount of time the member server will be offline updating its database points, we recommend that you use multiple save commands rather than a large configuration change with one save command. You can also minimize the number of changes that occur in a replication interval by modifying either the RepTransactionArchiveLimit or the RepTransactionSyncInterval, or both of these properties. For example, instead of using the default value of 100 for the RepTransactionArchiveLimit, you might change it to 20.

---

**Note**

The IP address format is enhanced to support both IPv4 and IPv6.

---

# How Replication Works

The following sections describe the flow of a simple replication as it occurs under normal conditions:

- **Replication Data Flow**
- **Security**
- **Replication Archive**
- **Ensuring Data Integrity**
- **Full Resynchronization**
- **Understanding Hot-Configuration**
Replication Data Flow

The following sections describe data flow on the master server and the slave server:

- **Master Server**
- **Slave Server**

**Master Server**

The master server or primary server is the fully configured machine that is used to archive all the transactions that taken place in Prime Access Registrar.

**Performing the Data Flow for the Master Server**

To perform data flow for the master server:

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The administrator makes a change to the master server’s configuration using the <code>aregcmd</code> command line interface (CLI) and issues a <code>save</code> command.</td>
</tr>
<tr>
<td>2</td>
<td>After the changes are successfully validated, the changes are stored in the Prime Access Registrar database.</td>
</tr>
<tr>
<td>3</td>
<td><code>aregcmd</code> then notifies the Prime Access Registrar server executing on the master of the configuration change.</td>
</tr>
<tr>
<td>4</td>
<td>The Prime Access Registrar server then updates its version of the configuration stored in memory. (This is called hot-config because it happens while the server is running and processing requests.)</td>
</tr>
<tr>
<td>5</td>
<td>The Prime Access Registrar server first copies the changes pertaining to the <code>aregcmd save</code>, also known as a transaction to its replication archive, then transmits the transaction to the slave server for processing.</td>
</tr>
<tr>
<td>6</td>
<td>In <code>aregcmd</code>, the prompt returns indicating that the <code>save</code> has completed successfully, the transaction has been archived, and the transaction has been transmitted to the slaves.</td>
</tr>
</tbody>
</table>

**Slave Server**

The slave server or secondary server is a fully-installed and configured system stands ready to takeover when the primary machine fails.

**Performing Data Flow for the Slave Server**

To perform data flow for the slave server:

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>When the slave server receives the transaction, its contents are verified.</td>
</tr>
<tr>
<td>2</td>
<td>After verification, the changes are applied to the slave server’s database.</td>
</tr>
<tr>
<td>3</td>
<td>The changes are then applied (hot-configured) in the slave server’s in-memory configuration.</td>
</tr>
<tr>
<td>4</td>
<td>The transaction is written to the slave server’s replication archive.</td>
</tr>
</tbody>
</table>
Security

Replication has two primary security concerns:

1. Security of the transactions transmitted to the slave server
2. Storage of transactions in the replication archive

Both of these concerns use shared secret (MD5) encryption via the shared secret specified in the replication configuration on both master and slave servers. Replication data transmitted between master and slave is encrypted at the source and decrypted at the destination the same way as standard RADIUS packets between Prime Access Registrar's clients and the Prime Access Registrar server. Transactions written to the replication archive are also encrypted in the same manner and decrypted when read from the replication archive.

Replication Archive

The replication archive serves two primary purposes:

- To provide persistent, or saved, information regarding the last successful transaction
- To persist transactions in case the slave server requires resynchronization (see Ensuring Data Integrity below for more information on resynchronization).

The replication archive is simply a directory located in ../CSCOar/data/archive. Each transaction replicated by the master is written to this directory as a single file. The name of each transaction file is of the form txn############ where ############ is the unique transaction number assigned by the master server. The replication archive size, that is the number of transaction files it might contain, is configured in the Replication configuration setting of TransactionArchiveLimit. When the TransactionArchive limit is exceeded, the oldest transaction file is deleted.

Ensuring Data Integrity

Prime Access Registrar's configuration replication feature ensures data integrity through transaction data verification, transaction ordering, automatic resynchronization and manual full-resynchronization. With the single exception of a manual full-resynchronization, each of the following techniques help to automatically ensure that master and slave servers contain identical configurations. A detailed description of each technique follows. This section contains the following topics:

- Transaction Data Verification
- Transaction Order
- Automatic Resynchronization

Transaction Data Verification

When the master prepares a transaction for replication to a slave, the master calculates a 2's complement Cyclic Redundancy Check (CRC) for each element (individual configuration change) in the transaction and for the entire transaction and includes these CRC values in the transmitted transaction. When the slave receives the transaction, the slave calculates a CRC for each transaction element and for the entire transaction and compares its own calculated values with those sent with the message. If a discrepancy occurs from these comparisons, the transaction element or the entire transaction is discarded and a re-transmission of that particular transaction element or the entire transaction is requested by the slave from the master. This process is called automatic resynchronization. (described in more detail below)
Transaction Order

When the master prepares a transaction for replication, it assigns the transaction a unique transaction number. This number is used to ensure the transactions are processed by the slave in exactly the same order as they were processed on the master. Transactions are order dependent. Since the functionality of Prime Access Registrar's configuration replication feature is to maintain identical configurations between master and slave, if transaction order were not retained, master and slave would not contain identical configurations. Consider where two transactions modify the same thing (a defined client's IP address for example). If the first transaction was a mistake and the second was the desired result, the client configuration on the master would contain the second setting; however, if the transactions were processed in the reverse order on the slave, the client configuration on the slave would contain the mistaken IP Address. This example illustrates the critical need for transaction ordering to ensure data integrity.

Automatic Resynchronization

Automatic Resynchronization is the most significant feature with respect to data integrity. This feature ensures the configurations on both the master and slave are identical. If they are not, this feature automatically corrects the problem.

When the master and slave start-up, they determine the transaction number of the last replication transaction from their respective replication archives. The master immediately begins periodic transmission of a TransactionSync message to the slave. This message informs the slave of the transaction number of the transaction that the master last replicated.

If the transaction number in the TransactionSync message does not match the transaction number of the last received transaction in the slave's archive, then the slave will request resynchronization from the master. The resynchronization request sent by the slave will include the slave's last received transaction number.

The master will respond by retransmitting each transaction since the last transaction number indicated by the slave in the resynchronization request. The master obtains these transactions from its replication archive.

Should the slave's last received transaction number be less than the lowest transaction number in the master's replication archive, then automatic resynchronization cannot occur as the master's replication archive does not contain enough history to synchronize the slave. In this case, the slave must be resynchronized with a full-resynchronization.

Full Resynchronization

Full Resynchronization means that the slave has missed more transactions than are stored in the master's replication archive and cannot be resynchronized automatically. There is no automatic full-resynchronization mechanism in Prime Access Registrar's configuration replication feature. To perform a full resynchronization, see Full Resynchronization.
Understanding Hot-Configuration

Hot-Configuration is the process of reflecting configuration changes made to Prime Access Registrar’s internal configuration database in the in-memory configuration of the executing Prime Access Registrar server. Hot-Configuration is accomplished without interruption of RADIUS request processing. For example, if an administrator uses aregcmd to configure a new client and issues a save command, when the prompt returns, the newly configured client can send requests to Prime Access Registrar.

Hot-Configuration minimizes the down-time associated with having to restart an Prime Access Registrar server to put configuration changes into effect. With the Hot-Configuration feature, a restart is only necessary when a Session Manager, Resource Manager or Remote Server configuration is modified. These configuration elements might not be hot-configured because they maintain state (an active session, for example) and cannot be modified without losing the state information they maintain. Changes to these configuration elements require a restart of Prime Access Registrar to put them into effect.

Hot-Configuration is not associated with the replication feature. Hot-Configuration’s only connection to the replication feature is that when a change is replicated to the slave, the slave is hot-configured to reflect the replicated change as if an administrator had used aregcmd to make the changes directly on the slave server.

Replication’s Impact on Request Processing

The replication feature was designed to perform replication of transactions with minimal impact on RADIUS request processing. When a transaction is received by a slave, RADIUS requests are queued while the transaction is applied to the slave. After the transaction is complete, RADIUS request processing resumes.

The impact on RADIUS request processing is a direct result of the size of a transaction. The smaller the transaction the lesser the impact, and the larger the transaction, the greater the impact. In other words, when making changes to the master, frequent saves are better than making lots of changes and then saving. Each change is one transaction element and all changes involved in a save comprise a single transaction with one element per change. Since the replication feature only impacts RADIUS request processing when changes are made, the impact under normal operation (when changes are not being made) is virtually unmeasurable.

Replication Configuration Settings

This section describes each replication configuration setting. In aregcmd, replication settings are found in //localhost/Radius/Replication. This section contains the following topics:

- RepType
- RepTransactionSyncInterval
- RepTransactionArchiveLimit
- RepIPAddress
- RepPort
- RepSecret
- RepIsMaster
- RepMasterIPAddress
- RepMasterPort
• Rep Members Subdirectory
• Rep Members/Slave1
• Name
• IPAddress
• Port

RepType

RepType indicates the type of replication. The choices available are SMDBR and NONE.
When RepType is set to NONE, replication is disabled. To enable replication, set RepType to SMDBR for Single Master Database Replication. RepType must be set to SMDBR on both the master and slave servers.

RepTransactionSyncInterval

Master

On the master server, RepTransactionSyncInterval is the duration between periodic transmission of the TransactionSync message expressed in milliseconds. The default is 60000 or 1 minute.
The purpose of RepTransactionSyncInterval is to indicate how frequently to check for an out-of-sync condition between the master and slave servers. When the slave received the TransactionSync message, it uses its contents to determine if it needs to resynchronize with the master.
The larger the setting for RepTransactionSyncInterval, the longer the period of time between out-of-sync detection. However, if RepTransactionSyncInterval is set too small, the slave can frequently request resynchronization when it is not really out of sync. If the duration is too small, the slave cannot completely receive a transaction before it receives the TransactionSync message. In this case, the servers will remain synchronized, but there will be unnecessary excess traffic that could affect performance.

Note

We recommend that you use smaller values for the RepTransactionSyncInterval to limit the time a slave server is offline applying change sets during automatic resynchronization.

Slave

On the slave, RepTransactionSyncInterval is used to determine if the slave has lost contact with the master and to alert administrators of a possible loss of connectivity between the master an slave. If the elapsed time since the last received TransactionSync message exceeds the setting of RepTransactionSyncInterval, the slave writes a log message indicating that it might have lost contact with the master. This log message is repeated each TransactionSyncInterval until a TransactionSync message is received.
RepTransactionArchiveLimit

On both master and slave, the RepTransactionArchiveLimit setting determines how many transactions can be stored in the archive. The default setting is 100. When the limit is exceeded, the oldest transaction file is deleted. If a slave requires resynchronization and the last transaction it received is no longer in the archive, a full resynchronization will be necessary to bring the slave back in sync with the master.

Note

The value set for RepTransactionArchiveLimit should be the same on the master and the slave.

An appropriate value for RepTransactionArchiveLimit depends upon how much hard disk space an administrator can provide for resynchronization. If this value is large, say 10,000, then the last 10,000 transactions will be stored in the archive. This is like saying the last 10,000 saves from aregcmd will be stored in the archive. Large values are best. The size of each transaction depends upon how many configuration changes were included in the transaction, so hard disk space usage is difficult to estimate.

Note

We recommend that you use smaller values for the RepTransactionArchiveLimit to limit the time a slave server is offline applying change sets during automatic resynchronization.

If the slave should go down or otherwise be taken off line, the value of RepTransactionArchiveLimit and the frequency of aregcmd saves will determine how long the slave can be offline before a full-resynchronization will be required.

There are two reasons why a slave server should have an archive:

1. The slave must save the last received transaction for resynchronization purposes (at a minimum).
2. Should the master go down, the slave can then be configured as the master and provide resynchronization services to other slaves.

RepIPAddress

The RepIPAddress value is set to the IP Address of the machine containing the Prime Access Registrar installation.

Note

The IP address format is enhanced to support both IPv4 and IPv6.

RepPort

The RepPort is the port used to receive of replication messages. In most cases, the default value (1812) is sufficient. If another port is to be used, the interfaces must exist in the machine.

RepSecret

RepSecret is the replication secret shared between the master and slave. The value of this setting must be identical on both the master and the slave.
RepIsMaster

The RepIsMaster setting indicates whether the machine is a master or a slave. On the master, set RepIsMaster to TRUE. On the slave set it to FALSE. Only the master can have this value set to TRUE and there can be only one master.

RepMasterIPAddress

RepMasterIPAddress specifies the IP Address of the master. On the master, set RepMasterIPAddress to the same value used in RepIPAddress above. On the slave, RepMasterIPAddress must be set to the IP Address of the master.

Note
The IP address format is enhanced to support both IPv4 and IPv6.

RepMasterPort

RepMasterPort is the port to use to send replication messages to the master. In most cases, the default value (1812) is sufficient; however, if another is to be used, the interfaces must exist in the machine.

Rep Members Subdirectory

The Rep Members\ subdirectory contains the list of slaves to which the master will replicate transactions.

Rep Members/Slave1

Each slave is added much like a client is added. Each slave must have a configuration in the Rep Members directory to be considered part of the replication network by the master. The master will not transmit any messages or replications to servers not in this list, and any communication received by a server not in this list will be ignored.

Note
Although it is possible to configure multiple slaves with the same master, we have only considered a single-master/single-slave configuration. This is the recommended configuration.

Name

This is the name of the slave. The name must be unique.

IPAddress

This is the IP Address of the slave.
Port

This is the port upon which the master will send replication messages to the slave.

Setting Up Replication

This section provides step-by-step instructions about how to configure replication on both the master and member servers. The “Replication Example” section on page 12-13, shows an example of replication configuration.

If possible, open an xterm window on both the master and member. In each of these windows, change directory to $INSTALL/logs and run xtail to watch the logs. This allows you to watch replication log messages as they occur. If you are using a system which had a previous installation of Prime Access Registrar, delete all files located in the $INSTALL/data/archive directory if it is present on either the master or member systems. This section contains the following topics:

- Configuring The Master
- Configuring The Member
- Verifying the Configuration

Configuring The Master

On the master server, RepTransactionSyncInterval is the duration between periodic transmission of the TransactionSync message expressed in milliseconds. The default is 60000 or 1 minute.

Configuring the Master Server for Replication

To configure the master server for replication:

**Step 1**
On the machine which is to be the master, using aregcmd, navigate to //localhost/Radius/Replication

**Step 2**
Set the RepType to SMDBR:

```plaintext
set RepType SMDBR
```

**Step 3**
Set the RepIPAddress to the IP address of the master:

```plaintext
set RepIPAddress 192.168.1.1
```

**Step 4**
Set the RepSecret to MySecret:

```plaintext
set RepSecret MySecret
```

**Step 5**
Set RepIsMaster to TRUE:

```plaintext
set RepIsMaster TRUE
```
Step 6  Set RepMasterIPAddress to the same value used in step 3:

\[ \text{set RepMasterIPAddress 192.168.1.1} \]

Step 7  Change directory to /Radius/Advanced and set the MaximumNumberOfRadiusPackets property to 8192:

\[ \text{cd /Radius/Advanced} \]

\[ \text{set MaximumNumberOfRadiusPackets 8192} \]

Step 8  Change directory to Rep Members:

\[ \text{cd “rep members”} \]

Note  You must enclose Rep Members in quotes due to the space in the name.

Step 9  Add member1:

\[ \text{add member1} \]

Step 10  Change directory to member1:

\[ \text{cd member1} \]

Step 11  Set the IPAddress to the IP Address of the machine to be the member:

\[ \text{set IPAddress 192.168.1.2} \]

Note  The RepPort and RepMasterPort properties on the Master must correspond to one of the ports configured in /Radius/Advanced/Ports, if one is configured. Otherwise, the default values for the RepPort and RepMasterPort properties are sufficient.

Step 12  Save the configuration:

\[ \text{save} \]

Step 13  Reload the configuration:

\[ \text{reload} \]

Configuring The Member

On the slave, RepTransactionSyncInterval is used to determine if the slave has lost contact with the master and to alert administrators of a possible loss of connectivity between the master and slave.
**Configuring the Member Server for Replication**

To configure the member server for replication:

---

**Step 1**
On the machine which is to be the member, using `aregcmd`, navigate to `//localhost/Radius/Replication`.

**Step 2**
Set the RepType to SMDBR.

```
set RepType SMDBR
```

**Step 3**
Set the RepIPAddress to the IP address of the member.

```
set RepIPAddress 192.168.1.2
```

**Step 4**
Set the RepSecret to MySecret.

```
set RepSecret MySecret
```

**Step 5**
Set RepMasterIPAddress to IP Address of the master (the same value used in Step 3 on page 8-1).

```
set RepMasterIPAddress 192.168.1.1
```

**Step 6**
Change directory to `/Radius/Advanced` and set the `MaximumNumberOfRadiusPackets` property to 8192.

```
cd /Radius/Advanced
set MaximumNumberOfRadiusPackets 8192
```

**Step 7**
If the Master has been configured to use a port other than the well-known (and default) RADIUS ports, configure each Member to use the same port.

**Note**
The RepPort and RepMasterPort properties on the Master must correspond to one of the ports configured in `/Radius/Advanced/Ports`, if one is configured. Otherwise, the default values for the RepPort and RepMasterPort properties are sufficient.

**Step 8**
Save the configuration:

```
save
```

**Step 9**
Reload the configuration:

```
reload
```

---

**Verifying the Configuration**

After both servers have successfully started, use `aregcmd` to make a small change to be replicated to the member server which you can easily verify. We recommend setting the description in `//localhost/Radius` to something like `Test1`. After you issue an `aregcmd save` and the prompt returns, run `aregcmd` on the member server and change directory to `//localhost/Radius`. Ensure that the description is set to `Test1`. If this was successful, then replication is properly configured and functional.
Replication Example

This section provides an example of replication and shows the actions that occur.

Adding a User

The Users object contains all of the information necessary to authenticate a user or authorize a user. Users in local UserLists can have multiple profiles. On the master server, use aregcmd to add a new user to the default user list.

Adding a New User
To add a new user:

---

**Step 1** Change directory to `/localhost/Radius/UserLists/Default`.
**Step 2** Enter the following:

```
add testuser
```

**Step 3** Change directory to `testuser`.

```
cd testuser
```

**Step 4** Set the password for `testuser`.

```
set password testuser
```

**Step 5** Confirm the password by entering `testuser` again.

**Step 6** Enter save to save the configuration.

---

Master Server’s Log

The log on the master shows the following:

```
*** ./name_radius_1_log ***
10/23/2013 23:17:07 name/radius/1 Info Server 0 Initiating Replication of Transaction 1 with 2 Elements.
10/23/2013 23:17:07 name/radius/1 Info Server 0 Replication Transaction #1 With 2 Elements Initiated
```

Member Server’s Log

The log on the member shows the following:

```
*** ./name_radius_1_log ***
10/23/2013 23:15:18 name/radius/1 Info Server 0 Radius Server is On-Line
10/23/2013 23:17:12 name/radius/1 Info Server 0 Committing Replication of Transaction 1 with 2 Elements.
10/23/2013 23:17:16 name/radius/1 Info Server 0 Replication Transaction #1 With 2 Elements Committed.
```
Verifying Replication

You can use one of two methods to verify that the new user testuser was properly replicated to the member:

- Run `aregcmd` on the member and look at the default userlist to see if it is there.
- Run `radclient` on the member and enter `simple testuser testuser` to create a simple access request packet (p001).
  
  Enter `p001 send` to send it. When it returns with p002, enter p002 to see if it is an Access Accept packet or an Access Reject packet. If it is an Access Accept, the user was properly replicated to the member. Using `radclient` is the recommended method to validate that a user was properly replicated.

On the Master, use `aregcmd` to delete the user from the default user list and save the user list.

Master Server’s Log

The log on the master shows the following:

```
*** ./name_radius_1_log ***
10/23/2013 23:20:48 name/radius/1 Info Server 0 Initiating Replication of Transaction 2 with 1 Elements.
10/23/2013 23:20:48 name/radius/1 Info Server 0 Replication Transaction #2 With 1 Elements Initiated
```

Member Server’s Log

The log on the member shows the following:

```
*** ./name_radius_1_log ***
10/23/2013 23:20:53 name/radius/1 Info Server 0 Committing Replication of Transaction 2 with 1 Elements.
10/23/2013 23:20:57 name/radius/1 Info Server 0 Replication Transaction #2 With 1 Elements Committed.
```

Repeat the validation procedure above to ensure the user `testuser` is no longer present on the member.

Using aregcmd -pf Option

Prime Access Registrar's replication feature works well using `aregcmd` input files. An `aregcmd` input file contains a list of `aregcmd` commands. For example, if the initial configuration of Prime Access Registrar were constructed in an input file, the master and member could be configured for replication first, then the input file applied to the master will be automatically replicated to the member.

Using aregcmd -pf Option

To illustrate replication using an `aregcmd` input file:

**Step 1** Create a text file called `add5users` with the following commands:

```
add /Radius/UserLists/Default/testuser1
cd /Radius/UserLists/Default/testuser1
```
set password testuser1
add /Radius/UserLists/Default/testuser2
cd /Radius/UserLists/Default/testuser2
set password testuser2
add /Radius/UserLists/Default/testuser3
cd /Radius/UserLists/Default/testuser3
set password testuser3
add /Radius/UserLists/Default/testuser4
cd /Radius/UserLists/Default/testuser4
set password testuser4
add /Radius/UserLists/Default/testuser5
cd /Radius/UserLists/Default/testuser5
set password testuser5
save

Step 2  On the master server, run the following command:

    aregcmd -pf add5users

---

Master Server’s Log

The log on the master shows the following:

*** ./name_radius_1_log ***
10/23/2013 23:27:08 name/radius/1 Info Server 0 Initiating Replication of Transaction 3 with 10 Elements.
10/23/2013 23:27:08 name/radius/1 Info Server 0 Replication Transaction #3 With 10 Elements Initiated

Member Server’s Log

The log on the member shows the following:

*** ./name_radius_1_log ***
10/23/2013 23:27:12 name/radius/1 Info Server 0 Committing Replication of Transaction 3 with 10 Elements.
10/23/2013 23:27:17 name/radius/1 Info Server 0 Replication Transaction #3 With 10 Elements Committed.

When the prompt returns, go to the member and use aregcmd to view the /radius/defaults/userlist. There should be five users there named testuser1 through testuser5.
An Automatic Resynchronization Example

This example will illustrate resynchronization of the member. This will be accomplished by stopping the member, making changes on the master, then restarting the member forcing a resynchronization.

Performing Resynchronization of the Member

To perform resynchronization of the member:

**Step 1**
At the member, stop the Prime Access Registrar server:

```
/etc/init.d/arservagt stop
```

At the master, run `aregcmd` and change directory to `/radius/userlist/default`.

```
cd /radius/userlist/default
```

**Step 2**
Enter the following:

```
add foouser
```

**Step 3**
Change directory to `foouser`.

```
cd foouser
```

**Step 4**
Set the password for `foouser`.

```
set password foouser
```

**Step 5**
Confirm the password by entering `foouser` again.

**Step 6**
Save the configuration:

```
save
```

Master Server's Log

The log on the master shows the following:

```
*** ./name_radius_1_log ***
10/23/2013 23:31:02 name/radius/1 Info Server 0 Initiating Replication of Transaction 5 with 2 Elements.
10/23/2013 23:31:02 name/radius/1 Info Server 0 Replication Transaction #5 With 2 Elements Initiated
```

On the member, run `/etc/init.d/arservagt start`. Notice the following log messages in the Master's log:

```
*** ./name_radius_1_log ***
10/23/2013 23:33:19 name/radius/1 Info Server 0 Resynchronizing member1.
```
Member Server’s Log

The log on the member shows the following:

```
*** ./name_radius_1_log ***
11/07/2013 23:33:14 name/radius/1 Info Server 0 Radius Server is Off-Line
11/07/2013 23:33:14 name/radius/1 Info Server 0 Starting Replication Manager
11/07/2013 23:33:24 name/radius/1 Info Server 0 Master Selected As Partner (DEFAULT)
11/07/2013 23:33:24 name/radius/1 Info Server 0 Radius Server is Off-Line
11/07/2013 23:33:24 name/radius/1 Warning Server 0 Requesting resynchronization from Master: Last Txn#3
11/07/2013 23:33:24 name/radius/1 Info Server 0 Resynchronization from Master in progress.
11/07/2013 23:33:24 name/radius/1 Info Server 0 Committing Replication of Transaction 4 with 2 Elements.
11/07/2013 23:33:28 name/radius/1 Info Server 0 Replication Transaction #4 With 2 Elements Committed.
11/07/2013 23:33:28 name/radius/1 Info Server 0 Radius Server is On-Line
```

As the log above shows, when the member started up, it validated its last received transaction number (#3) with the master's last replicated transaction number (#4). They did not match because a replication was initiated by the master which was not received by the member (because the member was stopped). When the member detected this discrepancy, the member made a resynchronization request to the master. The master responded by transmitting the missed transaction (#4) to the member. After it received and processed the retransmitted transaction, the member determined that it was then synchronized with the master and placed itself in an online status.

Full Resynchronization

Full Resynchronization means that the member has missed more transactions than are stored in the master's replication archive and can not be resynchronized automatically. There is no automatic full-resynchronization mechanism in Prime Access Registrar’s configuration replication feature. If a full resynchronization is required, you must export the master server’s database and update the member configuration.

Note

Before beginning, ensure there are no aregcmd sessions logged into the master server.

Performing a Manual Full-resynchronization

To perform a manual full-resynchronization:

Step 1

On the master server, stop the Prime Access Registrar server agent using the following command:

```
/etc/init.d/arserver stop
```

Step 2

On the master server, change directory to $INSTALL/data/db.

Step 3

Create a tarfile made up of the three database files, mcddb.d01, mcddb.d02, and mcddb.d03.

```
tar cvf /tmp/db.tar mcddb.d0*
```

Step 4

Create a tarfile of the archive.

```
tar cvf /tmp/archive.tar $INSTALL/data/archive
```

Step 5

On the master server, start the Prime Access Registrar server agent using the following command:
Step 6  On each member server requiring resynchronization, perform the following:

a. On the member server, stop the Prime Access Registrar server agent using the following command:

/etc/init.d/arserver stop

c. Change directory to $INSTALL/data/db, then untar the compressed database files.

cd $INSTALL/data/db
tar xvf /tmp/db.tar

d. Rebuild the key files using the following command:

$INSTALL/bin/keybuild mcdb

Note  This step might take several minutes.

e. Untar the archive.

cd $INSTALL/data/archive
tar xvf /tmp/archive.tar

f. As a safety check, run the following UNIX command to verify the integrity of the database.

$INSTALL/bin/dbcheck mcdb

Note  You must be user root to run dbcheck.

No errors should be detected.

h. Change directory to //radius/replication and change the following attributes:

- Change the RepIPAddress to that of the member.
- Change RepIsMaster to FALSE.
• Remove any entries under Rep Members.
  i. Save and reload the configuration.

  ```
save
Validating //localhost...
Saving //localhost...
```

  ```
reload
```

The member will start up and show online status in the log after it has verified it is synchronized with the master.

### Replication Setup with More Than One Slave

When replication is set up with more than one slave, Prime Access Registrar’s replication feature ensures that all the servers maintain identical configuration. This is done by forming a communication mesh. This mesh is formed by every server choosing two partners for itself from the replication setup. The servers tend to receive/send configuration updates from/to its partners. This ensures that all the servers maintain identical configuration inspite of minimal communication failures.

When bringing up a replication setup, Prime Access Registrar server comes up first and then initiates a partner sync request to all its replication members as visible from the configuration. So, a slave server will initiate partner sync to its master only. This is because master server is the only server visible to the slave server from the configuration. The master server will then broadcast the partner syncs that it has received, to all its replication members (slaves). Based on the sync messages sent by the master to this server, the evaluation of workload happens. The partner selection is based on the workload evaluation. Choosing the partners based on workload, ensures that the workload is equally distributed across the partner network.

The partners are selected based on the count of partner syncs received from the master:

- If partner syncs that have been received is one, choose the master as a partner.
- If partner syncs that have been received is two, choose the master and the other replication server as partners.
- If partner syncs that have been received is greater than two, perform a workload evaluation on the partners. Identify two servers that do not have two partners and choose them as partners.

### Frequently Asked Questions

**Question:** When I do a `save` in `aregcmd` and the validation fails, is anything replicated?

**Answer:** No; replication does not occur until `aregcmd` successfully saves the changes.

**Question:** Can I specify multiple masters with the same members?

**Answer:** No; the replication feature was designed to be used with a single-master. Also, it is not possible to specify more than one master in a member's configuration.

**Question:** Do I have to configure the master as a client on the member servers?
Answer: No. In-fact, it would be erroneous to do so. With the exception of Administrators, Interfaces, Replication, and Advanced machine-specific settings, the configuration between master and member must be identical. The replication feature's purpose is to maintain that relationship. Altering configuration settings on the member which are managed by the master will likely result in an unstable and possibly non-operational server.

Question: What configuration elements are replicated and what are not?
Answer: With the exception of Administrators, Interfaces, Replication, and Advanced machine-specific settings, all other settings are replicated.

Question: What configuration elements are hot-configured and what are not?
Answer: Session Managers, Resource Managers and Remote servers are not hot-configured because they maintain state, such as an active session, and cannot be manipulated dynamically.

Question: What is an appropriate TransactionSyncInterval setting?
Answer: This depends upon how long you want to allow an out-of-sync condition to persist. The shorter the interval, the more often an out-of-sync condition is checked. However, this results in added network traffic, additional processing by Prime Access Registrar and, if the interval is too small, frequent unnecessary resynchronization requests. The default value of 60,000 milliseconds (1 minute) is usually sufficient; however, values of as little as 10,000 milliseconds (10 seconds) have been tested and have worked well.

Question: What is an appropriate TransactionArchiveLimit setting?
Answer: This depends upon two things:
1. How much hard disk space you are willing to devote to transaction archive storage
2. How often your configuration is changed (a save is issued through Aregcmd).

If you have limited hard disk space, then perhaps smaller values (less than 1000) are appropriate; however if you have sufficient hard disk space, values of 10,000 or greater are better. The primary reason for this preference is to limit the possibility of a full-resynchronization being required. A full-resynchronization is required when the member has missed so many transactions that the master no longer contains all the transactions necessary to resynchronize the member. The greater the limit, the longer the member can be down without requiring a full-resynchronization.

Question: Can I specify a member in the member configuration?
Answer: Yes, and this is recommended. In the member's replication configuration Rep Members list, specify another server, perhaps one which can be used in-case of critical failure of the master. If the master suffers a catastrophic failure (a hard disk crash, for example) the member can be reconfigured to be the master simply by setting the RepIsMaster to TRUE and changing the MasterIPAddress to its own IP Address and the member specified in its Rep Members list will perform as the member. Because the member has an archive of transactions, the new member can be automatically resynchronized. If the archive limit on the new master has been exceeded (the transaction file txn0000000001 is no longer present in the new master's archive directory), then the new member will require a full-resynchronization. Setting the member up in this manner prevents down-time if the master fails and allows configuration changes to be made on the new master.

Question: How can I prevent a full-resynchronization from ever being necessary?
Answer: You can't, but you can limit the possibility by setting the TransactionArchiveLimit to a large value (greater than 10000). Another technique is to periodically check the archive when the master and member are synchronized. If the number of transaction files is approaching 10,000, then you can stop the master and member servers, delete all files in the replication archive, and restart the master and member. The only side effect is that if the master or member suffers a catastrophic failure, a full resynchronization will be required.
**Question:** Can I use the member to process RADIUS requests along with the master?

**Answer:** Yes, and this was one of the goals of the replication feature. Keep in mind that session information is not replicated between master and member. To use session management in this environment, use Prime Access Registrar’s central session manager.

## Replication Log Messages

This section contains typical replication log messages and explains what each means.

This section include the following topics:

- Information Log Messages
- Warning Log Messages
- Error Log Messages
- Log Messages You Should Never See

### Information Log Messages

**Info Message Starting Replication Manager**

Displayed at start-up and indicates the Replication Manager is configured and enabled. (RepType=SMDBR)

**Info Message Replication Disabled**

Displayed at start-up and indicates that Replication is not enabled. (RepType=NONE)

**Info Message Radius Server is On-Line**

Displayed by the member at start-up to indicate the member is synchronized with the master and processing RADIUS requests. It is also displayed after a successfully completed resynchronization. This message is never displayed on the master.

**Info Message Radius Server is Off-Line**

Displayed by the member at start-up to indicate the radius server is not processing RADIUS requests until it can ensure synchronization with the master. When this is displayed after startup, it indicates the member is no longer synchronized with the master and is directly associated with a resynchronization request to the master. This message is never displayed on the master.

**Info Message Resynchronizing <member name>**

Displayed by the master to indicate that it is resynchronizing the specified member (member).

**Info Message Resynchronization from Master in progress.**

Displayed by the member to indicate the master is in the process of resynchronizing it.

**Info Message Resynchronization complete.**

Displayed by the member to indicate the resynchronization has completed successfully.

**Info Message Resynchronization did not complete before timeout. Retrying.**
Indicates the master did not complete the resynchronization before the member expected it to complete and that the member is re-requesting resynchronization from the master for the remaining missed transactions.

**Info Message** Master Selected As Partner (DEFAULT)

Displayed by the member to indicate that it has selected the master as a partner after successfully getting connected with the master. Partner selection is performed after analyzing the replication workloads on other replication members.

**Info Message** Initiating Replication of Transaction <transaction #> with <# of elements> Elements.

Displayed by the master to indicate that it is beginning replication of a transaction to the member.

**Info Message** Replication Transaction #<transaction #> With # of elements Elements Initiated

Displayed by the master to indicate that it has completed sending the transaction to the member.

**Info Message** Committing Replication of Transaction #<transaction #> with # of elements Elements Committed

Displayed by the member to indicate that the transaction has been successfully processed.

**Info Message** Stopping Replication Manager

Displayed at shutdown by both the master and member to indicate the replication manager is being shut down.

**Info Message** Stopping Replication Manager - waiting for replication to complete...

Displayed by the member when a shutdown is attempted while received replications are being processed. After the replications are complete, the shutdown will complete.

**Info Message** Replication in progress. Please wait...

Periodically displayed while a shutdown is pending and replications are being completed.

**Info Message** Replication Manager Stopped

Displayed by both the master and member to indicate the replication manager has been successfully shutdown.

### Warning Log Messages

**Warning Message** Transaction Sync not received within configured TransactionSyncInterval. Communication with the Master may not be possible.

The member displays this log messages to indicate that it has not received a TransactionSync message from the master within its configured TransactionSync interval.

**Warning Message** TXN_SYNC Received by Master from unknown member <ip address>. Validation Failed

Displayed by the master when a TransactionSync message is received by the master. Since there can be only one configured master in a replication network, and the master is the only server who can send a TransactionSync message, this indicates there is another configured master in the replication network.
Warning Message TXN_SYNC Received from unknown Master <ip address>. Validation Failed
Displayed by the member to indicate that a TransactionSync message was received from a server not configured as its master.

Warning Message Requesting resynchronization from Master: Last Txn#<transaction#>
Displayed by the member to indicate that it is requesting resynchronization from the master. The LastTxn# is the last transaction number the member received and processed successfully.

Warning Message Resynchronization Request received from unknown member.
Displayed by the master when a resynchronization request is received by a member who is not listed in its /radius/replication/rep members configuration.

Warning Message Resynchronization of <member name> requires Full Resynchronization.
Displayed by the master to indicate that the member cannot be automatically resynchronized because its last transaction number is not within the configured history length of the archive (TransactionArchiveLimit). A manual resynchronization of the member is required to put the member back in-sync.

Warning Message MEMBER_SYNC Received from unknown Master at <ip address>. Validation Failed
Displayed by a member indicating that a master, other than its configured master, is requesting partnership.

Warning Message MEMBER_SYNC Received by Master from unknown member <ip address>. Validation Failed
Displayed by the master to indicate a member not listed in its /radius/replication/rep members configuration has requested partnership.

Warning Message TXN_EXPECT Received by Master from unknown <ip address>.
Displayed by the master to indicate it has received a transaction which originated from another illegal master.

Warning Message TXN_EXPECT Received from unknown Master <ip address>.
Displayed by the member to indicate it has received a transaction which originated from a master other than its configured master.

Warning Message TXN_EXPECT Broadcast failed.
Indicates that the master could not initiate a replication.

Warning Message DATA_SYNC Received by Master from unknown <ip address>
Displayed by the master to indicate that it received a replication transaction from another illegal master.

Warning Message DATA_SYNC Received from unknown <ip address>
Displayed by the member to indicate that a transaction was received from a server external to the replication network.

Error Log Messages

Error Message DATA_SYNC Validation failed - CRC Mismatch
Displayed by the member to indicate a received transaction element is invalid.

Error Message TXN_SYNC: Failed To Get Member Socket Handle.
TXN_SYNC: Failed to get master’s socket handle.
MEMBER_SYNC could not get socket handle
TXN_EXPECT: Failed to get socket handle.
DATA_SYNC could not get socket handle.
These messages indicate an invalid interface configuration in Cisco Access
Replication Log Messages

Registrar.
They could also be the result of specifying an invalid RepPort setting.
Failed To Create TXN_SYNC packet. (out of packets?)
Failed To Create TXN_SYNC packet.
MEMBER_SYNC Failed to create packet. (out of packets?)
TXN_EXPECT Failed to create packet. (out of packets?)
DATA_SYNC Create packet failed. (out of packets?)
DATA_SYNC Create packet failed.

These message indicate that a packet could not be created. This could be the result of a low memory condition or the result of the /Radius/Advanced/ MaximumNumberOfRadiusPackets setting being set too low

Error Message
TXN_SYNC validation failed - Internal error (pTxnSync=NULL).
MEMBER_SYNC validate failed - Internal Error (pMemberSync=NULL)
DATA_SYNC Validation Failed - Internal (pDataSync = NULL).
TXN_EXPECT Could not add new datablock to pending transaction queue.
Replication Member could not be added to member list.
Replication Member could not be added to member list.

These messages are the result of a failed memory allocation possibly due to an out of memory condition.

Error Message
DATA_SYNC Packet creation failed - Invalid ordinal.
Attempt To Replicate Transaction With Zero Elements.
Internal Error - Selected member not valid
Internal Replication Error ChangeType <change type> For <element path>
Internal error - Replication manager is invalid

These messages indicate an internal application failure.

Error Message
Cannot archive transaction datablock
Could not archive transaction

These messages are the result of a failed archive attempt. This could be the result of a low disk space condition.

Error Message
Could not commit transaction to MCD
Cannot Get Value For Unsupported DataType <data type id>
MCD Replication Cannot Delete Value <element path>
MCD Replication Cannot Delete Directory <element path>
MCD Replication Cannot Delete Value For <element path> With Unsupported DataType <data type id>
MCD Replication Cannot Create Dir For <element path>
MCD Replication Cannot Set Value For <element path>
MCD Replication Cannot Set Value For <element path>
MCD Replication Cannot Set Value For <element path>
MCD Replication Cannot Set Value For <element path>
MCD Replication Cannot Set Value For <element path> With Unsupported DataType
<data type id>
MCD Replication Cannot Set Value For <element path> With UNKNOWN DataType <data type id>

These messages are the result of a failed replication commit attempt.

Log Messages You Should Never See

The following list contains log messages which you should never see displayed in a log. If any of these messages are displayed in the log, contact Prime Access Registrar technical support for assistance.

Error Message
DATA_SYNC Received from non-partner <ip address>
DATA_RE_SYNC CRC mismatch. Replying with NAK
DATA_RE_SYNC Commit Failed. Replying with NAK
EVAL_SYNC Validation failed. <ip address> is not a Master or Member of the Replication network
EVAL_SYNC Received from unknown member.
PARTNER_SYNC Received from unknown member <ip address>.
EVAL_SYNC Failed to create packet. (out of packets?)
EVAL_SYNC Failed to create packet.
EVAL_SYNC Validation failed - Internal Error (pEvalSync=NULL).
PARTNER_SYNC Failed to get socket handle.
PARTNER_SYNC Failed to create packet. (out of packets?)
PARTNER_SYNC Failed to create packet.
DATA_RE_SYNC Can't get socket handle
DATA_RE_SYNC Failed to create packet (out of packets?)
DATA_RE_SYNC Failed to create packet
DATA_RE_SYNC Failed validation - Internal Error (pReSync = NULL)
DATA_RE_SYNC Cannot Set Value For <element path>
DATA_RE_SYNC Cannot Set Value For <element path>
DATA_RE_SYNC Cannot Set Value For <element path>
DATA_RE_SYNC Cannot Set Value For <element path>
DATA_RE_SYNC Cannot Set Value For <element path> With Unsupported DataType <data type id>
DATA_RE_SYNC Cannot Set Value For <element path> With UNKNOWN DataType <data type id>;
DATA_RE_SYNC Received by Master from unknown member <ip address>
DATA_RE_SYNC Received from unknown Master <ip address> DATA_RE_SYNC Reply received by Master from unknown Member <ip address>
Could not replicate data element to partners.
Could not replicate to partners - Invalid Ordinal.
Using On-Demand Address Pools

Cisco Prime Access Registrar (Prime Access Registrar) provides support for On-Demand Address Pools (ODAP). Using ODAP, the Prime Access Registrar server manages pools of addresses. Each pool is divided into subnets of various sizes, and the Prime Access Registrar server assigns the subnets to virtual home gateways (VHG) and Provider Edge (PE) routers. The VHG/PE router has one On-Demand Address Pool configured for each VPN supported by that VHG/PE.

Prime Access Registrar has been enhanced to make ODAP functionality more accessible and to enable ODAP requests and normal user authentication to occur on the same Prime Access Registrar server. To achieve this functionality, a new Cisco vendor script `CiscoWithODAPIncomingScript` was written to direct ODAP requests to particular services and session managers. `CiscoWithODAPIncomingScript` also provides the same functionality as the previous `CiscoIncomingScript`.

Additionally, Prime Access Registrar has a new vendor type, `CiscoWithODAP` which references `CiscoWithODAPIncomingScript` as its IncomingScript and references the existing script, `CiscoOutgoingScript`, as its Outgoing Script.

*Figure 13-1* shows a simple MPLS VPN network with two VHG/PE routers, VHG-1 and VHG-2. The Prime Access Registrar server allocates IP subnets to the VHGs by way of VRFs which contain the subnets and addresses (address space) available.
Chapter 13  Using On-Demand Address Pools

In Prime Access Registrar, the VRFs are configured as users in an ODAP-users list under /Radius/UserLists. The VRF name is set in IOS for the ODAP pool. When a VRF requests a pool of addresses, Prime Access Registrar directs the request to a Session-Manager configured with the name odap-<VRF name>. Prime Access Registrar also directs ODAP accounting requests to the service odap-accounting.

In the example network shown in Figure 13-1, the VRFs are configured with the following address spaces:

- **VRF-ISP1.com**—consists of the address range 10.255.0.0 - 10.255.255.255 divided among the following subnets:
  - 10.255.0.0/24
  - 10.255.1.0/24
  - ...
  - 10.255.255.0/24

- **VRF-ISP2.com**—consists of the address ranges 10.0.0.0 - 10.10.255.255 and 10.255.0.0 - 10.255.10.255 divided among the following subnets:
  - 10.0.0.0/16
  - 10.1.0.0/16
  - ...
  - 10.255.0.0/24
  - 10.255.1.0/24
  - ...
  - 10.255.10.0/24

**Note** VRF-ISPe.com requires two ResourceManagers because it has subnets of two different sizes.

- **VRF-ISP3.com**—consists of the address range 1172.21.0.0 - 172.21.255.255 divided among the following subnets:
  - 172.21.0.0/18
  - 172.21.64.0/18
  - 172.21.128.0/18
  and
  - 172.21.192.0/24
  - 172.21.193.0/24
  - ...
  - 172.21.255.0/24

**Note** VRF-ISP3.com requires two ResourceManagers because it also has subnets of two different sizes.
This chapter contains the following sections:

- Cisco-Incoming Script
- Vendor Type CiscoWithODAP
- Configuring Cisco Prime Access Registrar to Work with ODAP

Cisco-Incoming Script

The CiscoWithODAPIncomingScript makes ODAP functionality more accessible. This script eases the configuration required to enable ODAP requests and normal user authentication to occur on the same Prime Access Registrar server. CiscoWithODAPIncomingScript also provides the functionality of the original CiscoIncomingScript.

If the Prime Access Registrar server receives an ODAP request, the server sets the Session-Key from the AcctSessionID and sets the services and session managers.

If the Prime Access Registrar server receives a non-ODAP request, other scripts, rules or policies that you might already have in place on the Prime Access Registrar server handle these requests.

This section contains the following topics:

- How the Script Works
- CiscoWithODAPIncomingScript

How the Script Works

The following describes how the script CiscoWithODAPIncomingScript works:

1. The script examines the incoming NAS-Identifier sent by the client (VHG). If the NAS-Identifier does not equal odap-dhcp then this request is not an ODAP request. Since this is not an ODAP request, the script does not do any more ODAP-specific processing and just calls CiscoIncomingScript to allow that script to process the request. If this is an ODAP request, this script removes the NAS-Identifier attribute because it is no longer needed.

2. The script sets the Authentication-Service and the Authorization-Service to odap-users, and it sets the Accounting-Service to odap-accounting.

3. The Prime Access Registrar server sends the request to the appropriate Session Manager based on the username. Session Managers with odap-<username> must be created and configured in Prime Access Registrar.

4. The script then uses Session IDs to identify each ODAP request. The script uses the Acct-Session-Id attribute as the Session-Key.

CiscoWithODAPIncomingScript

The following is a Tcl script example of the script CiscoWithODAPIncomingScript.

Note CiscoWithODAPIncomingScript is written in C language. This example script is more easily understood in Tcl.
Vendor Type CiscoWithODAP

You must configure any Clients that might forward ODAP requests to the Prime Access Registrar server as being of Vendor CiscoWithODAP.

This vendor type references the new script, CiscoWithODAPIncomingScript, as its IncomingScript and references the existing script, CiscoOutgoingScript, as its OutgoingScript.

After setting Vendor to CiscoWithODAP, ODAP requests are directed to the AA service, set to odap-users, the accounting service is set to odap-accounting, and the Session Manager is set to odap-username, where username is filled from the request. The username received in the request is a VRF name, the request is directed to the appropriate Session Manager.

Note

The final line in the example above is not how the script really works because a Tcl script cannot call a C script. This is one reason why CiscoWithODAPIncomingScript was written in C.
Configuring Cisco Prime Access Registrar to Work with ODAP

This section provides information about how to configure Prime Access Registrar to work with ODAP.

Configuring Prime Access Registrar to work with ODAP

You must configure any Clients that might forward ODAP requests to the Prime Access Registrar server as being of Vendor CiscoWithODAP.

To configure Prime Access Registrar to work with ODAP:

Step 1 Create and configure an ODAP-users UserList. All ODAP users are configured under this UserList.

Step 2 Add all ODAP users to the ODAP-users UserList. Usernames must be of the form `<vrf name>` with the AllowNullPassword property set to TRUE.

Step 3 Create and configure a service for ODAP-users.

Step 4 Create and configure an ODAP accounting service. Set the accounting service Type to `file` and FilenamePrefix `odap-accounting`.

Step 5 Create a Session Manager for each of the VRFs. There must be a separate Session Manager for each VRF pool.

Step 6 Create and configure Resource Managers to be referenced by the Session Managers.

Note: Subnet pools of different sizes (different subnet masks) require separate Resource Managers.

Step 7 Configure the Session Managers with the Resource Managers.

Step 8 Configure any Clients that might send ODAP requests to Vendor type CiscoWithODAP.

Step 9 Save your configuration.

Configuring the ODAP Detailed Instructions

You must configure any Clients that might forward ODAP requests to the Prime Access Registrar server as being of Vendor CiscoWithODAP.

To configure Prime Access Registrar to work with ODAP:

Setting Up an ODAP UserList

Step 1 Create a UserList for ODAP users.

```
--> cd /radius/userlists
```

[ //localhost/Radius/UserLists ]
Adding ODAP Users

**Step 2** Add the ODAP users to the ODAP UserList and set the AllowNullPassword property to TRUE. Each user is a VRF name set for each ODAP client.

```bash
[ //localhost/Radius/UserLists/odap-users ]
Entries 0 to 0 from 0 total entries
Current filter: <all>
Name = odap-users
Description =

--> add vrf-ISP1.com

Added vrf-ISP1.com

--> add vrf-ISP2.com

Added vrf-ISP2.com

--> add vrf-ISP3.com

Added vrf-ISP3.com

--> ls

[ //localhost/Radius/UserLists/odap-users ]
Entries 1 to 3 from 3 total entries
Current filter: <all>
Name = odap-users
Description =
vrf-ISP1.com/
vrf-ISP2.com/
vrf-ISP3.com/
```

**Step 3** Set the AllowNullPassword property to TRUE for each ODAP user.

```bash
--> cd vrf-ISP2.com

[ //localhost/Radius/UserLists/odap-users/vrf-ISP2.com ]
Name = vrf-ISP2.com
Description =
Password =
Enabled = TRUE
```
Setting Up an ODAP-Users Service

**Step 4** Add and configure a service for ODAP Users.

```bash
---> cd /radius/services

[ //localhost/Radius/Services ]
  Entries 1 to 2 from 2 total entries
  Current filter: <all>

  local-file/
  local-users/

---> add odap-users

Added odap-users

---> cd odap-users

[ //localhost/Radius/Services/odap-users ]
  Name = odap-users
  Description =
  Type =
  IncomingScript- =
  OutgoingScript- =

---> set type local

Set Type local

---> set userlist odap-users

Set UserList odap-users

---> ls

[ //localhost/Radius/Services/odap-users ]
  Name = odap-users
  Description =
  Type = local
  IncomingScript- =
  OutgoingScript- =
  OutagePolicy- = RejectAll
  OutageScript- =
  UserList = odap-users
```

```bash
---> set AllowNullPassword TRUE
```
Setting Up an ODAP Accounting Service

**Step 5**  Add and configure an ODAP accounting service.

```bash
-> cd /radius/services

[ //localhost/Radius/Services ]
Entries 1 to 3 from 3 total entries
Current filter: <all>

local-file/
local-users/
odap-users/

-> add odap-accounting

Added odap-accounting

-> cd odap-accounting

[ //localhost/Radius/Services/odap-accounting ]
Name = odap-accounting
Description =
Type =
IncomingScript- =
OutgoingScript- =

-> set type file

Set Type file

-> ls

[ //localhost/Radius/Services/odap-accounting ]
Name = odap-accounting
Description =
Type = file
IncomingScript- =
OutgoingScript- =
OutagePolicy- = RejectAll
OutageScript- =
FilenamePrefix = accounting
MaxFileSize = "10 Megabytes"
MaxFileAge = "1 Day"
RolloverSchedule =

-> set FilenamePrefix odap-accounting

Set Filenameprefix odap-accounting
```

Adding Session Managers

**Step 6**  Create one Session Manager for each of the VRF pools.

Create one Session Manager for each of the users you specify in the odap-users UserList. The Session Managers must be called odap-VRF_name to meet the requirements of CiscoWithODAPIncomingScript.
Setting Up Resource Managers

Step 7 Set up subnet-dynamic Resource Managers that are to be referenced by the Session Managers.

Session Managers can manage multiple Resource Managers. One or more subnet pools can be set up of varying sizes to allocate the ranges of subnet addresses you have available. Subnets of different sizes require different Resource Managers.

```
-- cd /radius/sessionmanagers

[ //localhost/Radius/SessionManagers ]
Entries 1 to 1 from 1 total entries
Current filter: <all>

session-mgr-1/

-- add odap-vrf-ISP1.com

Added odap-vrf-ISP1.com

-- add odap-vrf-ISP2.com

Added odap-vrf-ISP2.com

-- add odap-vrf-ISP3.com

Added odap-vrf-ISP3.com
```

Setting Up Resource Managers

-- cd /radius/resourcemanagers

```
[ //localhost/Radius/ResourceManagers ]
Entries 1 to 5 from 5 total entries
Current filter: <all>

IPA-Pool/  
IPA-Pool-2/  
IPX-Pool/  
Per-Group/  
Per-User/  

-- add odap-vrf-ISP1.com
```

Note

The names of Resource Managers do not have to be related to VRFs.

```
Added odap-vrf-ISP1.com

-- cd odap-vrf-ISP1.com

[ //localhost/Radius/ResourceManagers/odap-vrf-ISP1.com ]
Name = odap-vrf-ISP1.com
Description =
Type =

-- set type subnet-dynamic
```
Set Type subnet-dynamic

--> ls

[ //localhost/Radius/ResourceManagers/odap-vrf-ISP1.com ]
Name = odap-vrf-ISP1.com
Description =
Type = subnet-dynamic
NetMask =
SubnetAddresses/

-> set netmask 255.255.255.0

Set NetMask 255.255.255.0

--> cd subnetaddresses

[ //localhost/Radius/ResourceManagers/odap-vrf-ISP1.com/SubnetAddresses ]
Entries 0 to 0 from 0 total entries
Current filter: <all>

--> add 10.255.0.0-10.255.255.255

Added 10.255.0.0-10.255.255.255

Note

Two Resource Managers are required for VRF-ISP3.com and VRF-ISP2.com because their address spaces are made up of subnets of the different sizes.

--> cd /radius/resourcemanagers

[ //localhost/Radius/ResourceManagers ]
Entries 1 to 5 from 5 total entries
Current filter: <all>

IPA-Pool/
IPA-Pool-2/
IPX-Pool/
odap-vrf-ISP1.com/
Per-Group/
Per-User/

--> add odap-vrf-ISP3-a.com

Added odap-vrf-ISP3-a.com

--> cd odap-vrf-ISP3-a.com

Name = odap-vrf-ISP3-a.com
Description =
Type =

--> set type subnet-dynamic

Set Type subnet-dynamic
Chapter 13  Using On-Demand Address Pools

Configuring Cisco Prime Access Registrar to Work with ODAP

```bash
--> ls
    Name = odap-vrf-ISP3-a.com
    Description =
    Type = subnet-dynamic
    NetMask =
    SubnetAddresses/

--> set netmask 255.255.192.0
Set NetMask 255.255.192.0

--> cd subnetaddresses
    Entries 0 to 0 from 0 total entries
    Current filter: <all>

--> add 171.21.0.0-172.21.191.255
Added 172.21.0.0-172.21.191.255

--> cd /radius/resourcemanagers
[ //localhost/Radius/ResourceManagers ]
    Entries 1 to 10 from 10 total entries
    Current filter: <all>
    IPA-Pool/
    IPA-Pool-2/
    IPX-Pool/
    odap-vrf-ISP1.com/
    odap-vrf-ISP3-a.com /
    Per-Group/
    Per-User/

--> add odap-vrf-ISP3-b.com
Added odap-vrf-ISP3-b.com

--> cd odap-vrf-ISP3-b.com
[ //localhost/Radius/ResourceManagers/odap-vrf-ISP3-b.com ]
    Name = odap-vrf-ISP3-b.com
    Description =
    Type =

--> set type subnet-dynamic
Set Type subnet-dynamic

--> ls
[ //localhost/Radius/ResourceManagers/odap-vrf-ISP3-b.com ]
    Name = odap-vrf-ISP3-b.com
    Description =
    Type = subnet-dynamic
```
NetMask = SubnetAddresses/

-> set netmask 255.255.255.0

Set NetMask 255.255.255.0

-> cd subnetaddresses

[ //localhost/Radius/ResourceManagers/odap-vrf-ISP3-b.com /SubnetAddresses ]
Entries 0 to 0 from 0 total entries
Current filter: <all>

--> add 172.21.191.0-172.21.255.255

Added 172.21.191.0-172.21.255.255

-> cd /radius/resourcemanagers

[ //localhost/Radius/ResourceManagers ]
Entries 1 to 10 from 10 total entries
Current filter: <all>

IPA-Pool/
IPA-Pool-2/
IPX-Pool/
odap-vrf-ISP1.com/
odap-vrf-ISP3-a.com /
odap-vrf-ISP3-b.com /
Per-Group/
Per-User/

--> add odap-vrf-ISP2-a.com

Added odap-vrf-ISP2-a.com

--> cd odap-vrf-ISP2-a.com

Name = odap-vrf-ISP2.com
Description =
Type =

--> set type subnet-dynamic

Set Type subnet-dynamic

--> ls

Name = odap-vrf-ISP2-a.com
Description =
Type = subnet-dynamic
NetMask = SubnetAddresses/

-> set netmask 255.255.0.0

Set NetMask 255.255.0.0
-> cd subnetaddresses

Entries 0 to 0 from 0 total entries
Current filter: <all>

--> add 10.0.0.0-10.10.255.255

Added 10.0.0.0-10.255.255.255

-> cd /radius/resourcemanagers

[ //localhost/Radius/ResourceManagers ]
Entries 1 to 10 from 10 total entries
Current filter: <all>

IPA-Pool/
IPA-Pool-2/
IPX-Pool/
odap-vrf-ISP1.com/
odap-vrf-ISP3-a.com /
odap-vrf-ISP3-b.com /
odap-vrf-ISP2-a.com /
Per-Group/
Per-User/

--> add odap-vrf-ISP2-b.com

Added odap-vrf-ISP2-b.com

--> cd odap-vrf-ISP2-b.com

[ //localhost/Radius/ResourceManagers/odap-vrf-ISP2-b.com ]
Name = odap-vrf-ISP2-b.com
Description =
Type =

--> set type subnet-dynamic

Set Type subnet-dynamic

--> ls

[ //localhost/Radius/ResourceManagers/odap-vrf-ISP2-b.com ]
Name = odap-vrf-ISP2-b.com
Description =
Type = subnet-dynamic
NetMask =
SubnetAddresses/

--> set netmask 255.255.255.0

Set NetMask 255.255.255.0

-> cd subnetaddresses

[ //localhost/Radius/ResourceManagers/odap-vrf-ISP2-b.com /SubnetAddresses ]
Entries 0 to 0 from 0 total entries
Configuring Cisco Prime Access Registrar to Work with ODAP

Chapter 13      Using On-Demand Address Pools

Configuring Session Managers

Note: It is not necessary to configure Session Managers in two instances. All SessionManager configuration can be done at one time before configuring the Resource Managers.

Step 8 Configure the Session Managers to be referenced by the Resource Managers.

--> cd/radius/sessionmanagers

[ //localhost/Radius/SessionManagers ]
Entries 1 to 4 from 4 total entries
Current filter: <all>

odap-vrf-ISP1.com/
odap-vrf-ISP2.com/
odap-vrf-ISP3.com/
session-mgr-1/

--> cd odap-vrf-ISP2.com

[ //localhost/Radius/SessionManagers/odap-vrf-ISP2.com ]
Name = odap-vrf-ISP2.com
Description =
AllowAccountingStartToCreateSession = FALSE
ResourceManagers/.

--> cd resourcemanagers

--> set 1 odap-vrf-ISP2-a.com
Set 1 odap-vrf-ISP2-a.com

--> set 2 odap-vrf-ISP2-b.com
Set 2 odap-vrf-ISP2-b.com

--> cd/radius/sessionmanagers

[ //localhost/Radius/SessionManagers ]
Entries 1 to 4 from 4 total entries
Current filter: <all>

odap-vrf-ISP1.com/
odap-vrf-ISP2.com/
odap-vrf-ISP3.com/
session-mgr-1/

--> cd odap-vrf-ISP3.com
Chapter 13      Using On-Demand Address Pools

Configuring Cisco Prime Access Registrar to Work with ODAP

[ //localhost/Radius/SessionManagers/odap-vrf-ISP3.com ]
Name = odap-vrf-ISP3.com
Description =
AllowAccountingStartToCreateSession = FALSE
ResourceManagers/

--> cd resourcemanagers

--> set 1 odap-vrf-ISP3-a.com

Set 1 odap-vrf-ISP3-a.com

--> set 2 odap-vrf-ISP3-b.com

Set 2 odap-vrf-ISP3-b.com

--> cd/radius/sessionmanagers

[ //localhost/Radius/SessionManagers ]
Entries 1 to 4 from 4 total entries
Current filter: <all>
  odap-vrf-ISP1.com/
  odap-vrf-ISP2.com/
  odap-vrf-ISP3.com/
  session-mgr-1/

--> cd odap-vrf-ISP1.com

[ //localhost/Radius/SessionManagers/odap-vrf-ISP1.com ]
Name = odap-vrf-ISP1.com
Description =
AllowAccountingStartToCreateSession = FALSE
ResourceManagers/

--> cd resourcemanagers

--> set 1 odap-vrf-ISP1.com

Set 1 odap-vrf-ISP1.com

Configure Clients

Step 9  For any client that might forward ODAP requests to the Prime Access Registrar server, set the Vendor property to CiscoWithODAP.

--> cd /radius/clients

[ //localhost/Radius/Clients ]
Entries 1 to 2 from 2 total entries
Current filter: <all>
  localhost/
  vhg-1/
  vhg-2/

--> cd vhg-1
Chapter 13 Using On-Demand Address Pools

Saving //localhost...

Save Your Configuration

**Step 10** After completing the configuration, save your changes.

--> save

Validating //localhost...

Set Vendor CiscoWithODAP

```plaintext
[ //localhost/RADIUS/Clients/vhg-1 ]
  Name = vhg-1
  Description =
  IPAddresses = 209.165.200.225
  SharedSecret = secret
  Type = NAS
  Vendor =
  IncomingScript =
  OutgoingScript =
  USEDNIS = FALSE
  DeviceName = a_name
  DevicePassword = password

---> set vendor CiscoWithODAP

Set Vendor CiscoWithODAP
```
Using Identity Caching

Cisco Prime Access Registrar (Prime Access Registrar) software includes the identity caching feature. Prime Access Registrar runs as application layer software and can be used standalone or in conjunction with other workstations running Prime Access Registrar.

Note

The identity caching feature is available on Prime Access Registrar releases 3.5.2 and above.

Identity caching provides subscriber identity resolution services with fast access to associated subscriber identity data for service providers, enabling them to offer new services to their customers based on identity caching and context information management.

This chapter contains the following sections:

- Overview
- Identity Caching Features
- Configuring Cisco Prime Access Registrar for Identity Caching
- Starting Identity Caching

Overview

Identity caching enables Cisco equipment to gain context information about the operator’s subscribers to support network functions or to enhance subscriber’s experience on the operator’s network. Figure 14-1 on page 14-2, Prime Access Registrar System Overview, shows the network environment where Prime Access Registrar identity caching might be used.

For example, Client Services Gateway (CSG) uses IP mapping information provided by identity caching to support post-paid content billing. Identity caching acquires subscriber information from other devices and information sources in the operator’s network. The type of information provided is limited by the available information sources and is configurable by the operator, but might include information such as IP address, MSISDN, and IMSI. Identity caching does not duplicate the operator’s persistent data stores. Identity caching provides a protocol-based interface through which Cisco network elements (Prime Access Registrar identity caching clients) can access subscriber information.

The Prime Access Registrar servers receive RADIUS flows from the Gateway GPRS support Node (GGSN) which acts as a type of network access station (NAS). These flows perform full AAA (authentication, authorization, and accounting). You can configure the Prime Access Registrar servers to redirect the accounting information (only) to an identity caching server to be cached. The GGSN can also be configured to direct only the RADIUS accounting information directly to the Prime Access Registrar server.
Prime Access Registrar also receives XML identity query flows from the CSM which acts as a NAS. In the event that a CSM should fail or lose its information, the information can be refreshed from the information cached in the Prime Access Registrar server.

Prime Access Registrar acquires subscriber information such as the IP address, the mobile Subscriber ISDN number (MSISDN), and the International Mobile System Identifier (IMSI) from AAA requests the Prime Access Registrar server receives, typically from the GGSN. The types of information provided is limited by the available information sources and is configurable by the operator.

Prime Access Registrar includes an XML Query Identity enhancement. Prime Access Registrar previously supported User-Name lookup based on the Framed IP address of an existing session. The XML Query Identity enhancement enables Framed IP address lookup based on the User-Name in an existing session.

![Prime Access Registrar System Overview](image)

**Figure 14-1  Prime Access Registrar System Overview**

**Identity Caching Features**

Prime Access Registrar identity caching provides the following features:

- Supports GGSN subscriber data attributes from RADIUS authentication sequences
- Provides basic identity mapping services from IP address or username/APN to Mobile DN for one network presence at a time.
- Provide session management support for Content Switch Module (CSM)

Prime Access Registrar enables the CSM to keep the data session and content correlated to the same subscriber reconnecting, perhaps after an attach/detach sequence for a GPRS subscriber connecting again. This is done through the MSISDN identity to IP mapping in the identity caching function.
• Enhance redundancy with stateful fail-over support for applications by finding the right connection between subscriber identity and IP address using the Identity Cache function.

• Uses an XML interface to make it easier for any network function or application to use without having to have detailed internal knowledge about the execution environment or programming methods.

• Provides user identity resolution with fast access to associated subscriber data

• Establishes an identity and Access Management solution that can be used in and across multiple network domains

• Provides a way to use identity resolution to manage the growth of 2.5G mobile data access services (GSM/GPRS) and to provide always-on mobile data access including the following:
  – Ties various IP addresses to a unique subscriber identifier
  – Dynamically assigning and reusing IP addresses and controlling services with consistent identification
  – Correlates previous content activity when a mobile subscriber reconnects
  – Correlates IP addresses, mobile numbers, username, and identifiers to support customer billing
  – Correlates and identifies subscribers using both 2.5G and WLAN services and provides a way to control and manage operator network services
  – Provides subscriber privacy control
  – Provides a way to cache content with various customers and their networks

Configuring Cisco Prime Access Registrar for Identity Caching

Use the command line interface aregcmd to configure Prime Access Registrar to perform identity caching.

Configuring the Identity Caching

To configure identity caching:

**Step 1**
Launch aregcmd.

**Step 2**
Define a client object for each client that will send either RADIUS or XML packets to the Prime Access Registrar server performing identity caching.

There should be one client object for each GGSN, one for each CSM and one for each packet simulator (if used in a test environment).

For example, if a packet simulator will be used on the same server where you perform identity caching, add a client object as in the following:

```bash
cd /Radius/Clients
add xml-client
cd xml-client
```
This client object is very similar to the localhost object defined in the example configuration. The \texttt{SharedSecret} property will be ignored if the client is an XML client, but still must be set to a non-null value. The \texttt{Type} property is also ignored for XML clients.

**Step 3** Define a port object for each RADIUS port and each XML port to be used. Two RADIUS ports, the second immediately following the first in numeric value, must be defined even if only one is needed. A typical identity caching installation requires the following port configuration:

```
cd /Radius/Advanced/Ports
add 1812
add 1813
add 8080
```

**Note** Although ports 1812 and 1813 are the default ports for Prime Access Registrar, you must add them to \texttt{/Radius/Advanced/Ports} to also add port 8080.

**Step 4** Change directory to the 1812 port and set its type to Radius-Access.

```
  cd /Radius/Advanced/Ports/1812
  set Type Radius-Access
```

**Step 5** Change directory to the 1813 port and set its type to Radius-Accounting.

```
  cd /Radius/Advanced/Ports/1813
  set Type Radius-Accounting
```

**Step 6** Change directory to the 8080 port and set its type to XML.

```
  -cd /Radius/Advanced/Ports/8080
  set Type XML
```

**Step 7** Define and configure an accounting service of type file and set it as the DefaultAccountingService. An accounting service is required for Prime Access Registrar to cache identity information, even if no accounting service is needed otherwise. If you added the example configuration during installation, a local-file accounting service is already configured.

If you did not add the example configuration during software installation, see the Setting Up Accounting section in Chapter 7, “RADIUS Accounting.”
Chapter 14      Using Identity Caching

Configuring Cisco Prime Access Registrar for Identity Caching

Step 8  Define and configure a ResourceManager for identity caching.

    cd /Radius/ResourceManagers

    add cache

Step 9  Set the ResourceManager to type session-cache for identity caching.

    cd cache

    set type session-cache

The following shows the default properties of a session-cache ResourceManager:

    [ //localhost/Radius/ResourceManagers/cache ]
    Name = cache
    Description =
    Type = session-cache
    OverwriteAttributes = FALSE
    QueryKey =
    PendingRemovalDelay = 10
    AttributesToBeCached/
    QueryMappings/

Step 10  Set the QueryKey to a RADIUS attribute you want to key on.

For example, use the following command to set the QueryKey to User-Name:

    set QueryKey User-Name

The QueryKey must match the string on the right-hand side of one of the pairs you list in QueryMappings. It is not necessary for the QueryKey to be configured under AttributesToBeCached because the QueryKey will always be cached by default.

Note  The QueryKey property must always be a RADIUS attribute. The Prime Access Registrar server forces a NULL IP address (0.0.0.0) if it detects an incorrectly configured QueryKey.

Step 11  Change directory to AttributesToBeCached and use the set command to provide a list of RADIUS attributes you want to store in cache.

    cd AttributesToBeCached

    set 1 Calling-Station-ID
    Set 2 User-Name
    Set 3 Framed-IP-Address

The attributes a session-cache resource manager caches can be queried through both RADIUS Query and XML Query packets. When you cache attributes Framed-IP-Address or User-Name, or when you use XML-Address-format-IPv4 or XML-UserId-id_type-subscriber_id as the QueryKey, you must map the XML attributes to RADIUS attributes in the QueryMappings subdirectory.
Step 12  Change directory to QueryMappings and use the set command to list the attribute pairs, mapping the XML attributes on the left-hand side to the RADIUS attribute on the right-hand side.

```
set XML-Address-format-IPv4 Framed-IP-Address
set XML-UserId-id_type-subscriber_id User-Name
```

Step 13  Change directory to /Radius/SessionManagers and add a SessionManager for identity caching.

```
cd /Radius/SessionManagers
add IDcache
```

Step 14  Change directory to the new identity caching SessionManager, then change directory to the ResourceManager list.

```
cd IDcache/ResourceManagers
```

Step 15  Use the set command to associate the identity caching ResourceManager with this SessionManager.

```
set 1 cache
```

Step 16  Change directory to /Radius and set the DefaultSessionManager to the identity caching SessionManager.

```
cd /Radius
set DefaultSessionManager IDcache
```

Step 17  Run the save, reload, and exit commands:

```
save
reload
exit
```

---

Starting Identity Caching

To start identity caching, you must send an Accounting-Request to the specified accounting port (The default accounting port is 1813.) A minimal Accounting-Request will contain the following attributes:

- NAS-Identifier or NAS-IP-Address
- NAS-Port
- Framed-IP-Address
- User-Name
- Acct-Status-Type
- Acct-Session-Id
**Starting Identity Caching**

To start identity caching:

---

**Step 1**

Launch `radclient`:

```
cd /opt/CSCOar/bin

radclient -C localhost -N admin -P aicuser
```

**Step 2**

Enter the following `radclient` commands:

```
set p [ acct_request Start joeuser@cisco.com ]
$p set attrib [ attrib Framed-IP-Address 123.123.123.123 ]
$p send
```

This assumes that you are running `radclient` on the same server and using 1813 as the accounting port.

**Step 3**

Send XML requests to the specified XML port (Cisco suggests port 8080 as shown above). A typical XML packet will look like the following:

```xml
<?xml version="1.0"?>
<Request>
  <UserIdRequest>
    <UserId id_type="subscriber_id">bob</UserId>
  </UserIdRequest>
</Request>
```

To do this using `xmlclient`, put the XML text into a file, then enter the following command:

```
cd /opt/CSCOar/bin

./xmlclient -srd <file>
```

---

**Note**

This assumes that xmlclient is running on the same server as identity caching and that 8080 is the XML port. Use the command `xmlclient -H` for information about how to use a different port or how to run `xmlclient` from a different server.

---

**Note**

For a successful query, xml response will have the IPAddress associated with the requested user-name and for failure query it returns 0.0.0.0 as the IPAddress.
XML Interface

The XML interface is used for subscriber context information queries and responses to those queries. The XML interface is on a UDP port (8080) and is configurable. Identity caching supports the XML data-type definition (DTD) supported by the CSG.

The mapping from queries to replies can be one to many. For example, a UDP datagram might contain several queries but each reply will be returned in a separate datagram. No single query or reply can exceed the configured MTU of a datagram. Any that does results in an error.

If a query result is negative, the reply will consist of a null subscriber ID. All other error conditions cause Prime Access Registrar to drop the request. Errors are logged locally using the Prime Access Registrar logging mechanism.
Using Trusted ID Authorization with SESM

Cisco Prime Access Registrar (Prime Access Registrar) can be used in a Service Selection Gateway (SSG) - Cisco Subscriber Edge Services Manager (SESM) deployment to enable the Trusted Identity (Trusted ID) Authorization feature. This chapter describes how to use Prime Access Registrar with SESM, and how to configure Prime Access Registrar to use the Trusted ID feature.

The Trusted ID feature provides transparent login capabilities for users based on a trusted ID instead of the user's name, enabling end users of an SSG to maintain an always-on connection without the need to authenticate on each connect. Using SSG's Transparent Auto-Login (TAL) feature, a TAL access-request packet contains a Trusted ID, such as a MAC address, that identifies the user without the user's real username and password. The SESM Profile Management Guide provides detailed information about Trusted ID authorization in SESM.

If Prime Access Registrar knows the user associated with the Trusted ID, Prime Access Registrar uses the Trusted ID to authenticate and authorize the user. If the authentication and authorization succeeds, Prime Access Registrar returns the user's username in the Access-Accept so the SSG can include the user's identity in subsequent Accounting-Requests.

If Prime Access Registrar does not know the user associated with the Trusted ID, Prime Access Registrar returns an Access-Reject. The Access-Reject causes the SSG to redirect the user to a SESM web portal login page. When the user explicitly authenticates, Prime Access Registrar captures the Trusted ID and maps it to a user association so subsequent attempts to authenticate with the Trusted ID succeed.

This chapter contains the following sections:

- Trusted ID Operational Overview
- Software Requirements
- Configuring Cisco Prime Access Registrar for Trusted Identity with SESM
- Configuration Imported by TrustedIdInstall Program
- Configuring EAP-MD5 Authentication

**Trusted ID Operational Overview**

This section describes the following operations of the Trusted ID Authentication feature:

- Configuration Overview
- Request Processing
- Session Cache Life Cycle
Configuration Overview

The Trusted ID features require two objects in Prime Access Registrar, a UserService, a SessionManager, and a ResourceManager. The UserService references another service called to perform the authentication and authorization (AA). The SessionManager references a SessionManager that contains a reference to a session-cache ResourceManager. These objects are imported into the Prime Access Registrar server configuration when you run the TrustedIdInstall.bin program.

Configuration Imported by TrustedIdInstall Program, page 15-13 lists the configuration imported into the Prime Access Registrar server by the TrustedIdInstall.bin program.

The Resource Manager is configured with the QueryKey property set to a RADIUS attribute that contains the Trusted ID such as the Calling-Station ID. The Query Key should be set to an attribute present in all appropriate AA requests that uniquely identifies the user such as Calling-Station ID. The Query Key can be set to only one RADIUS attribute.

The Resource Manager is also configured to cache the attributes required to identify the user, username, and the user’s credentials, password or CHAP-Password and CHAP-Challenge. The attributes User-Name, User-Password, NAS-Identifier, NAS-Port, or NAS-Port-Type are not appropriate choices for Query Key because they do not uniquely identify users.

The RollingEncryptionKeyChangePeriod specifies the length of time a given EncryptionKey will be used before a new one is created. When the session-cache ResourceManager caches User-Password attributes, Prime Access Registrar encrypts the User-Password so it is not stored in memory or persisted on disk in clear text. Prime Access Registrar uses up to 255 encryption keys, using a new one after each RollingEncryptionKeyChangePeriod expires. If RollingEncryptionKeyChangePeriod is set to 2 days, Prime Access Registrar will create and begin using a new EncryptionKey every two days. The oldest key will be retired, and Prime Access Registrar will re-encrypt any User-Passwords that used the old key with the new key. This way, if the RollingEncryptionKeyChangePeriod is set to 1 day, no key will be older than 255 days.

The encryption keys are indirectly connected to Trusted ID. Since User-Passwords might be stored for a long time in memory and on disk, Prime Access Registrar uses the RollingEncryptionKey to encrypt the User-Passwords. The RollingEncryptionKey makes it more difficult for someone to crack or decode the User-Passwords because the key used changes frequently. If someone were to break one key, that would only give them the ability to decrypt those User-Passwords that had been encrypted with that key. All others, including those yet to be encrypted after the key change period expires would not be vulnerable.

Request Processing

When the Trusted ID service processes Access-Requests, it queries the session-cache ResourceManager for a cache entry associated with the Trusted ID. If found, the Resource Manager returns the cached attributes. The Trusted ID service replaces the request’s existing attributes with the cached attributes.

After the Resource Manager is queried (and the request’s existing attributes are replaced with the cached attributes if the cache entry exists), the Trusted ID’s UserService authenticates and authorizes the request. The UserService is always called whether the cache entry exists or not. The only attributes cached in the Resource Manager are the ones listed in AttributesToBeCached. The user profile is usually not cached and is retrieved each time by the UserService.
Whether the request succeeds or not, the request is passed on to the service referenced by the UserService property. When that service completes authentication and authorization, control returns to the Trusted ID service. The session-cache might be updated if AA is successful.

**Session Cache Life Cycle**

Session cache management comprises adding and deleting Trusted ID to user mapping to and from the cache and is initiated from the Trusted ID service. The mapping is one-to-one mapping. For each Trusted ID, there can be only one cache entry, and conversely for each cache entry, there can be only one Trusted ID.

If a user is not presently in the session cache (the query failed), the AA done by theUserService succeeded and the internal attribute (Implicit-Auth-Enabled) was returned with a value of `true`, Prime Access Registrar adds the user to the cache. Since the AA succeeded, Prime Access Registrar assumes this is an explicit authentication by the user and the attributes required by the session-cache are present in the Access-Request.

If the user is already in the session cache (the query succeeded) and the AA done by the UserService failed, the internal attributes Implicit-Auth-Enabled was not returned, or was returned with a value other than true, Prime Access Registrar removes the user from the session cache.

If the user has enabled implicit authentication (and if that results in Implicit-Auth-Enabled being returned as `true`), after the first Explicit Auth (from the login page), the user will be in the cache and will always be implicitly authenticated and authorized. In this case, you can get them out of the cache three ways:

- Have the user disable implicit authentication, then reconnect
- Have the system administrator release the session using `aregcmd` commands
- Use the SessionTimeout property in the Session Manager

If the user’s account becomes orphaned (the user no longer exists), the cache entry will persist until it is removed using `aregcmd`.

If you have disabled implicit authentication, you are forced to authenticate each time and the cache is not updated. If you subsequently enable implicit authentication, you must explicitly authenticate one more time to create the user’s cache entry. After creating the user’s cache entry, they will not need to explicitly authenticate again (with this instance of Prime Access Registrar) as long as implicit authentication is enabled.

**Configuration Restrictions**

The Session Manager referenced by the TrustedID Service should not be used for general session management. The Trusted ID Session Manager should be a separate Session Manager used only for the Trusted ID session cache. The data in the session-cache must persist longer than the length of the session. If the Trusted ID Session Manager was used for general session management, the cache would be updated for the general session, overwriting the cache entry for the special session created for the Trusted ID service. When the general session ended it would delete that data and subsequent queries for implicit authentication would fail.
Software Requirements

The Trusted ID feature requires the following software to be installed:

- Cisco Subscriber Edge Services Manager (SESM) 3.3(1)
- Cisco Subscriber Policy Engine (SPE) 2.1.12
- Cisco Prime Access Registrar

In addition to the software listed above, you must run **TrustedIdInstall.rpm**, a Java application that runs on the Linux platform.

**Note**

The disk space required to run the **TrustedIdInstall** program is about 1.3 MB.

The **TrustedIdInstall** program verifies the software prerequisites, installs the required jar files, and extends the configuration for Prime Access Registrar. The **TrustedIdInstall** program is only available on Cisco.com under the Prime Access Registrar download area at the following URL:

[http://www.cisco.com/cgi-bin/tablebuild.pl/access-registrar-encrypted](http://www.cisco.com/cgi-bin/tablebuild.pl/access-registrar-encrypted)

This section contains the following topics:

- Installing Cisco Prime Access Registrar
- Running the TrustedIdInstall Program

## Installing Cisco Prime Access Registrar

See the *Cisco Prime Access Registrar 6.1.1 Installation and Configuration Guide* for detailed information about how to install Prime Access Registrar software.

**Note**

You must specify a Java Runtime Environment (JRE) when you install Prime Access Registrar software.

## Running the TrustedIdInstall Program

Cisco provides a Java-based program called **TrustedIdInstall** that installs required jar files, the configuration for Subscriber Policy Engine (SPE), and Prime Access Registrar. The **TrustedIdInstall** program can be run as an InstallShield wizard using the graphical user interface (GUI) or from the command line.

**Note**

Before running the TrustedIDInstall program, ensure that the SPE 2.1.12 software has been installed with SESM 3.3(1) (in SPE mode).

## Using the TrustedIdInstall.bin GUI

You must run the **TrustedIdInstall** program on the workstation where Prime Access Registrar is installed with a Java Runtime Environment (JRE) up to and including 1.4.2 in the path.
Chapter 15  Using Trusted ID Authorization with SESM

Software Requirements

Installing the TrustedIdInstall
To install TrustedIdInstall:

Step 1  Log in as a user with root privileges.
Step 2  Enter the following from the Prime Access Registrar server’s command line:

```
Step 2
```

```
TrustedIdInstall.rpm
```

The following message appears after you enter the command line above:

```
# TrustedIdInstall.rpm
InstallShield Wizard

Initializing InstallShield Wizard...
Searching for Java(tm) Virtual Machine...
............running under 1.2
```

Figure 15-1 shows the welcome window of the Trusted ID Azn AR SESM Integration 1.0 Installer.

![Figure 15-1 Trusted ID Azn AR SESM Integration 1.0 Installer Welcome](image)

Step 3  Click Next to continue.

Step 4  Check to ensure that Cisco SESM 3.3(1) is installed and available on the network, then click Next.

Step 5  Select the vendor name of the LDAP data store you are using for SPE, then click Next.

The InstallIdInstall.rpm wizard displays the Password Encryption Panel. This panel prompts you for a master password (entered twice to ensure accuracy) and a Password Encryption Algorithm which can be None, SHA, or SHA-1.
Software Requirements

Note
If you plan to use EAP-MD5 authentication, choose None. See Configuring EAP-MD5 Authentication, page 15-15 for information about configuring EAP-MD5 authentication.

Step 6
Enter the password in field provided, and select the password encryption type, then click Next.

Step 7
If in Step 5 you selected iPlanet as the Data Store Type, continue with Step 8. If you chose any other Data Store Type, proceed to Step 9.

The iPlanet Data Store Type requires that you set the value for the naming variable in ACNSchema.xml and DESSSchema.xml, either for Uid or Cn as shown in Figure 15-2. You can set the naming variable to either Uid or Cn.

Figure 15-2 Selecting iPlanet Naming Variable

Step 8
Select either Uid or Cn as the inetOrgPerson naming variable, then click Next.

The InstallIdInstall.rpm wizard displays the Service Type Selection panel.

Step 9
Accept the default Trusted ID Service Enable True or click to select False, then click Next.

The TrustedIdInstall program displays a panel that indicates the following:
- Location where the Trusted ID Authorization SESM Integration files will be stored (/cisco-ar)
- Features to be stored (Admin Tool)
- Amount of space required (about 1.3 MB)

The InstallIdInstall.rpm wizard displays the Directory Information panel, requesting information about the directory server required to extend the schema.

Step 10
Provide the requested Directory Server information as shown in Figure 15-3.
Chapter 15  Using Trusted ID Authorization with SESM

Software Requirements

Figure 15-3  Directory Server Information

Contact the directory administrator if you are unsure about the information required.

a. Enter a Directory Address.
   The Directory Address field requires the directory server IP address or DNS hostname.

b. Enter a Directory Port number.
   Provide the TCP/IP port on which your directory server listens. (This is usually port 389.)

c. Enter a Directory Admin User.
   Provide the User ID of the directory server administrator with permissions to extend the schema in the form:
   cn=admin

d. Enter a Directory Admin Password.
   Provide the password for the directory administrator user.

e. Enter a Directory Container.
   Provide the container in which the default RBAC objects should be created in the form:
   ou=sesm,o=cisco

f. Enter a DESS Admin User.
   Provide the User ID of the DESS administrator in the form:
   uid=admin,ou=sesm,o=cisco

g. Enter a DESS Admin Password.
   Provide the password for the DESS administrator.

Step 11  Click Next to continue.

The InstallIdInstall.rpm wizard begins the installation and displays a progress bar. When the installation completes, the wizard displays any warnings or errors it might have detected. Both boxes being empty indications a successful install.
Step 12  Click Next to continue.  
A final window indicates a successful installation of the Trusted ID Authorization AR SESM Integration software.

Step 13  Click Finish.

---

**Using the TrustedIdInstall Command Line**

You can run the **TrustedIdInstall** program using the command line option on a workstation where Prime Access Registrar is installed with a JRE up to and including 1.4.2 in the path. The command line interface requires the same information as the GUI method.

**Note**  
You must be a root user to run the **TrustedIdInstall** program

**Installing the TrustedIdInstall using Command Line**  
To install TrustedIdInstall using command line:

**Step 1**  
To run the **TrustedIdInstall** program using the command line interface, enter the following from the Prime Access Registrar server’s command line:

```
TrustedIdInstall.rpm -console
```

```
InstallShield Wizard

Initializing InstallShield Wizard...

Searching for Java (tm) Virtual Machine...
............
-------------------------------------------------------------------------------
Welcome to the InstallShield Wizard for Trusted ID Azn AR SESM Integration.  
The InstallShield Wizard will install Trusted ID Azn AR SESM Integration on your computer.  
To continue, choose Next.  
Trusted ID Azn AR SESM Integration1.0  
Cisco Systems, Inc.  
http://www.cisco.com
```

Press 1 for Next panel, 3 to Cancel or 4 to Redisplay [1] 1

The line above provides a way for you to enter your selection. You can press **Enter** to go to the next panel. Enter 3 to cancel the installation, or enter 4 to redisplay the current panel.

**Step 2**  
Press **Enter** to go to the next panel.

Please read the information below.

Cisco Systems  
Prerequisites  
Please ensure that minimally the following products are installed.  
1 Check to ensure that Cisco SESM 3.3(1) is installed and available on the network  
2 Checking for Prime AR 6.0 or later
Please ensure the configuration parameter supplied during SESM installation is used in this integration.

Press 1 for Next panel, 2 for Previous panel, 3 to Cancel or 4 to Redisplay (1) 1

This panel lists prerequisites required for successful installation. Before continuing to the next panel, ensure that SESM 3.3(1) is installed and available or the network. The program checks for Prime Access Registrar 3.5.3 (or later).

**Step 3** After insuring that SESM 3.3(1) is installed and available on the network, press **Enter**.

[X] 1 - Novell Directory Server
[ ] - iPlanet
[ ] - Data Communications Directory
[ ] - IBM Directory Server
[ ] - Open LDAP

Choose the Vendor for Directory, Select 0 to exit [0]

Press 1 for Next panel, 2 for Previous panel, 3 to Cancel or 4 to Redisplay [1]

This panel requests the data store type selection and indicates the Novell Directory Server is the default selection.

**Step 4** Press **Enter** to select the Novell Directory Server.

You can press 2 to select iPlanet, 3 to select Data Communications Directory, 4 to select IBM Directory Server, or 6 to select Open LDAP.

Enter the master password for SPE

Master Password []

This panel requests a master password for SPE.

**Step 5** Enter a password to be used as the master password for SPE and press **Enter**.

You are asked to re-enter the master password. The following panel requests an encryption algorithm and generates a secret key using the master password and selected algorithm.

[X] 1 - NONE
[ ] - SHA
[ ] - SSHA

Choose the installation type for SPE, Select 0 to exit [0]

Press 1 for Next panel, 2 for Previous panel, 3 to Cancel or 4 to Redisplay [1] 1

This panel indicates the default installation type as None. Enter “2” and press **Enter** to select SHA, or enter “3” and press **Enter** to select SSHA.

**Note** If you plan to use EAP-MD5 authentication, choose **None**. See Configuring EAP-MD5 Authentication, page 15-15 for information about configuring EAP-MD5 authentication.
Step 6  If in Step 4 you selected iPlanet as the Data Store Type, continue with Step 7. If you chose any other Data Store Type, proceed to Step 8.

+---------------+---------+
|              |        |
|   Uid        | Cn     |
+---------------+---------+

The iPlanet Data Store Type requires that you set the value for the naming variable in ACNSchema.xml and DESSSchema.xml, either for Uid or Cn as shown above.

Step 7  Press Enter to use the naming variable to Uid, or press 2 to select Cn.

Service Type Selection panel

Trusted ID Service Enable
[X]  1 - True
[ ]  2 - False

To select a choice enter its number, or 0 when you are finished [0]:

Press 1 for Next panel, 2 for Previous panel, 3 to Cancel or 4 to Redisplay [1] 1

The Service Type Selection panel asks if you want to enable the Trusted ID service. Enter 2 to choose to not enable the Trusted ID service.

Step 8  Press Enter to enable the Trusted ID service.

This panel indicates the location where the TrustedIdInstall program will write data and the amount of storage required.

Step 9  Press Enter to begin writing data.

Enter the IP Address (or) hostname of the system where the directory server is running. Please contact your directory administrator if you are not sure about this information.

Please enter the host address [localhost]:

Step 10  Press Enter to use the current system as the directory server, or enter another directory server name or IP address.

Enter the TCP/IP Port on which your directory server listens. Usually, the port is 389. Please contact your directory administrator if you are not sure about this information.

Please enter the Port number [389]:

Software Requirements
Step 11 Press **Enter** to use the default port, 389, or enter a different port number.

**Note**
Contact your directory server administrator if you are not sure about which port to use or other information required in the following steps.

Enter the User Id of the directory server with permissions to extend schema. Please contact your directory administrator if you are not sure about this information.

Please enter directory user
[uid=admin,ou=Administrators,ou=TopologyManagement,o=NetscapeRoot]:

Step 12 Enter the User ID of the directory server administrator with the necessary permissions to extend the schema.

Enter the password for the above user. Please contact your directory administrator if you are not sure about this information.

Please enter the password []: cisco

Step 13 Enter the password for the user provided in the previous step.

Enter the container in which the default RBAC objects should be created. Please contact your directory administrator if you are not sure about this information.

Please enter the container [o=cisco]:

Step 14 Press **Enter** to use the default container, or enter a different container and press **Enter**. Enter the User Id of the DESS user.

Please enter Dess user [cn=dessadmin,o=cisco]:

Step 15 Press **Enter** to use the default DESS user, or enter a different user ID and press **Enter**. Enter the password of the DESS user. Please contact your directory administrator if you are not sure about this information.

Please enter the Dess user password []: cisco

Step 16 Enter the DESS user password, then press **Enter**. Press 1 for Next panel, 3 to Cancel or 4 to Redisplay [1] 1

At this point, the software installation is ready to begin.
Configuring Cisco Prime Access Registrar for Trusted Identity with SESM

Use the command line interface `aregcmd` to configure Prime Access Registrar to use Trusted ID authorization in SSG-SESM deployments.

This section contains the following topics:
- Configuring the RADIUS Ports
- Configuring NAS Clients
- Configuring AAA and SPE Services

Configuring the RADIUS Ports

By default, Prime Access Registrar listens on ports 1812 and 1813 for any type of RADIUS request. It might be necessary to change the port assignments in the case of a resource collision. For example, if the RADIUS Directory Enabled Service Selection (DESS) Proxy (RDP) component of SPE is using ports 1812 and 1813, a port assignment change would be required.

The following command sequence causes Prime Access Registrar to listen on the explicitly defined ports, 1812 and 1813, for all types of RADIUS requests.

```
    cd /Radius/Advanced/Ports

    add 1812 ""radius
    Added 1812

    add 1813 ""radius
    Added 1813
```

After changing the port assignments, Prime Access Registrar no longer listens on the default ports. It might be necessary to add ports 1812 and 1813 if you are also using Prime Access Registrar for other AAA functionality.
Configuring NAS Clients

Change directory to `/Radius/Clients`, then add and configure the NAS clients required by SESM deployments:

```
change directory to /Radius/Clients
add SESM1 "" 10.3.3.2 cisco
Added SESM1
add SESM2 "" 10.3.3.101 cisco
Added SESM2
add SESM3 "" 10.3.3.102 cisco
Added SESM3
```

Configuring AAA and SPE Services

To configure AAA and SPE services:

```
Step 1 Change directory to `/Radius/Services`, then add and configure an accounting service.
   change directory to /Radius/Services
   add SESMaccounting "" file
   Added SESMaccounting

Step 2 Change directory to `/Radius`, then configure a DefaultAccountingService.
   change directory to /Radius
   set DefaultAccountingService SESMaccounting
   Set DefaultAccountingService SESMaccounting
```

Configuration Imported by TrustedIdInstall Program

The following is a listing of the configuration imported into the Prime Access Registrar server when you run the TrustedIdInstall program:
Chapter 15
Using Trusted ID Authorization with SESM

Configuration Imported by TrustedIdInstall Program

- /Radius
- /radius/services/spe
- /radius/services/trusted-id
- /Radius/SessionManagers/session-cache/
- /radius/ResourceManagers/session-cache
- /radius/advanced/
- /Radius/Scripts/ChangeServiceType

/Radius

DefaultAuthenticationService trusted-id
DefaultAuthorizationService trusted-id

/radius/services/spe

type java
ClassName com.cisco.cns.security.arspe.SPEExtension

/radius/services/trusted-id

type trusted-id
UserService spe
SessionManager session-cache

/Radius/SessionManagers/session-cache/

AllowAccountingStartToCreateSession FALSE
ResourceManagers/1 session-cache

/radius/ResourceManagers/session-cache

type session-cache
OverwriteAttributes TRUE
PendingRemovalDelay 10
QueryKey Calling-Station-ID
AttributesToBeCached/1 User-Name
AttributesToBeCached/2 User-Password
AttributesToBeCached/3 Calling-Station-ID

/radius/advanced/

ClasspathForJavaExtensions /cisco-ar/conf
Configuring EAP-MD5 Authentication

EAP-MD5 authentication is an optional authentication configuration. The following configuration changes are required to support EAP-MD5 authentication:

- Creating the CheckEap.tcl Script
- Adding the CheckEap.tcl Script
- Using the CheckEap.tcl Script
- Adding the EAP-MD5 Authentication Service
- Adding an LDAP Remote Server
- Adding an LDAP Service
- Saving the Configuration and Reloading the Server
- Cisco SSG VSAs in Cisco Prime Access Registrar Dictionary

**Note**

If you configure Prime Access Registrar to use EAP-MD5 authentication with the Trusted ID feature, you will not be able to use the Transparent Auto Login feature.

Creating the CheckEap.tcl Script

The **CheckEap.tcl** script must be created and stored in the file called `/cisco-ar/scripts/radius/tcl/CheckEap.tcl`. Use a text editor and copy the following lines into the CheckEap.tcl file:

```tcl
proc CheckEap { request response environment } {
    if { ![ $request containsKey EAP-Message ] } {
        $environment put Authentication-Service "EAP-MD5"
        $environment put Authorization-Service "spe"
    }
}
```
Chapter 15      Using Trusted ID Authorization with SESM

Adding the CheckEap.tcl Script

To add the CheckEap.tcl script:

Step 1  Start aregcmd, then change directory to /Radius/Scripts and add the CheckEap script.

```
cd /Radius/Scripts
add EapCheck
```

Step 2  Change directory to EapCheck.

```
cd EapCheck
[ //localhost/Radius/Scripts/EapCheck ]
   Name = EapCheck
   Description =
   Language =
```

Step 3  Set the Language property to TCL.

```
set Language TCL
```

Step 4  Set the filename property to CheckEap.tcl.

```
set Filename CheckEap.tcl
```

Step 5  Set the EntryPoint property to CheckEap.

```
set EntryPoint CheckEap
```

Note  The following sections also require you to use aregcmd, the command line interface.

Using the CheckEap.tcl Script

This section describes how to configure Prime Access Registrar to use the CheckEap script by setting the /Radius/IncomingScript property to CheckEap.

```
cd /Radius
set IncomingScript EapCheck
```
Adding the EAP-MD5 Authentication Service

To add and configure the EAP-MD5 service:

Step 1 Change directory to `/Radius/Services` and add an EAP-MD5 service.

```bash
cd /Radius/Services
add EAP-MD5
```

Step 2 Change directory to the EAP-MD5 service and set the Type and UserService properties as shown below:

```bash
cd EAP-MD5
```

Step 3 Change directory to the EAP-MD5 service.

```bash
cd EAP-MD5
```

Step 4 Set the service Type property to eap-md5 and the UserService property to LDAP.

```bash
set Type eap-md5
set UserService LDAP
```

The following example shows the configuration of the EAP-MD5 service:

```
[ //localhost/Radius/Services/EAP-MD5 ]
Name = EAP-MD5
Description =
Type = eap-md5
IncomingScript~ =
OutgoingScript~ =
AuthenticationTimeout = 120
UserService = LDAP
```

Adding an LDAP Remote Server

Prime Access Registrar adds a new type of service and remote server called ldap-accounting that enables inserting accounting records into LDAP. You can write accounting records into LDAP by referring this service in `/Radius/DefaultAccountingService` or in the Accounting-Service environment variable.

Adding and Configuring an LDAP Remote Server

To add and configure an LDAP remote server:

Step 1 Change directory to `/Radius/RemoteServers` and add a RemoteServer object.

```bash
cd /Radius/RemoteServers
add LDAP
```

Step 2 Change directory to the LDAP RemoteServer.


```
cd LDAP

[ //localhost/Radius/RemoteServers/LDAP ]
Name = LDAP
Description =
Protocol =

Step 3  Set the RemoteServer protocol property to ldap.

  set Protocol ldap

The following example shows the default configuration of an LDAP remote server:

[ //localhost/Radius/RemoteServers/LDAP ]
Name = LDAP
Description =
Protocol = ldap
Port = 389
ReactivateTimerInterval = 300000
Timeout = 15
HostName =
BindName =
BindPassword =
UseSSL = FALSE
SearchPath~ =
Filter~ = (uid=%s)
UserPasswordAttribute = userpassword
LimitOutstandingRequests = FALSE
MaxOutstandingRequests = 0
MaxReferrals = 0
ReferralAttribute =
ReferralFilter =
PasswordEncryptionStyle = Dynamic
EscapeSpecialCharInUserName = FALSE
DNSLookupAndLDAPRebindInterval =
LDAPToRadiusMappings/
LDAPToEnvironmentMappings/
LDAPToCheckItemMappings/

Step 4  Set the HostName property to the SPE/DESS directory IP address or hostname.

Step 5  Set the BindName property to the SPE/DESS administrator name.

Step 6  Set the BindPassword property to the SPE/DESS administrator password.

Step 7  Set the SearchPath property to the SPE/DESS directory container.

Step 8  Set the UserPasswordAttribute property type to clearpassword.
```

---

**Adding an LDAP Service**

You must configure a service of type ldap-accounting under /Radius/Services using the ldap accounting feature.

**Adding and Configuring an LDAP Service**

To add and configure an LDAP service:

---

**Step 1** Change directory to /Radius/Service and add LDAP.
cd /Radius/Service

add LDAP

**Step 2** Change directory to LDAP and set the type property to ldap.

cd LDAP

set Type ldap

The following shows the default configuration for an LDAP service:

```
[ //localhost/Radius/Services/LDAP ]
Name = LDAP
Description =
Type = ldap
IncomingScript- =
OutgoingScript- =
OutagePolicy- = RejectAll
OutageScript- =
MultipleServersPolicy = Failover
RemoteServers/
```

**Step 3** Change directory to RemoteServers and associate the LDAP RemoteServer with the LDAP service.

cd RemoteServers

set 1 LDAP

---

**Saving the Configuration and Reloading the Server**

Use the `save` command to save the configuration, then `reload` the Prime Access Registrar server.

```bash
save
Validating //localhost...
Saving //localhost...
```

```bash
reload
Reloading Server 'Radius'...
Server 'Radius' is Running, its health is 10 out of 10
```
Cisco SSG VSAs in Cisco Prime Access Registrar Dictionary

The following vendor-specific attributes (VSAs) are defined by default in the attribute dictionary after installing Prime Access Registrar software:

- Cisco-AVPair
- Cisco-SSG-Account-Info
- Cisco-SSG-Service-Info
- Cisco-SSG-Command-Code
- Cisco-SSG-Control-Info
Using Prepaid Billing

Cisco Prime Access Registrar (Prime Access Registrar) supports two types of prepaid billing, IS835C and Cisco Real-time Billing (CRB), a Cisco proprietary solution. The IS835C version adheres to industry standards and is the preferred version.

Three components are required to support a prepaid billing service, such as the following:

- AAA client
- Prime Access Registrar server
- External prepaid billing server

The most important factor for an effective prepaid billing service is in developing a shared library to be configured under the prepaid RemoteServer object. The shared library should be developed to implement all specified API functions. You will have to provide a shared library that meets the needs of your environment. The shared library must implement the API functions to perform the various tasks required for your specific implementation of the prepaid billing service.

Note

Cisco works with you to develop the prepaid billing service and implement the API. For more information, contact your Cisco systems engineer.

The chapter contains the following sections:

- Overview
- IS835C Prepaid Billing
- CRB Prepaid Billing
- Implementing the Prepaid Billing API
Overview

When a subscriber uses a prepaid billing service, each call requires a set of data about the subscriber. However, the AAA network has no previous knowledge of the subscriber’s usage behavior. Prime Access Registrar uses an iterative authorization paradigm over multiple sessions to support the prepaid billing solution.

Each time an authorization request is made, the billing server apportions a fraction of the subscriber’s balance into a quota. When a subscriber uses multiple sessions, each session must obtain its own quota. When a previously allocated quota is depleted, a session must be reauthorized to obtain a new quota.

The granularity and the magnitude of the quota is in the design and implementation of the prepaid billing server and is beyond the scope of this document. In general, a smaller quota generates more network traffic, but allows more sessions per subscriber. When the quota is equal to a subscriber’s total account balance, there is minimal network traffic, but only one session can be supported.

When a subscriber’s current quota is depleted, the AAA client initiates a reauthorization request sending Access-Request packets. After the Prime Access Registrar server receives the request, it forwards the request to the billing server. The billing server then returns the next quota to use. The new quota might not be the same as the previous, and the billing server might adjust the quota dynamically.

IS835C Prepaid Billing

Prime Access Registrar acts as a RADIUS protocol head for all the requirements specified in the cdma2000 Wireless IP Network Standard: PrePaid Packet Data Service specification:

http://www.3gpp2.org/Public_html/specs/X.S0011-006-C-v1.0.pdf

As long as the prepaid client understands or accepts what the external billing server sends, the service should work. The Prime Access Registrar server neither imposes nor is affected by the values of attributes returned from the external billing server.

For additional information, see cdma2000 Wireless IP Network Standard: Accounting Services and 3GPP2 RADIUS VSAs at the following URL:

http://www.3gpp2.org/Public_html/specs/X.S0011-005-C-v1.0.pdf

The IS835C specification requires that the Prime Access Registrar server be able to determine that a particular user is a prepaid billing user. A user is accepted as a valid prepaid user when the response dictionary of the incoming packet contains the Prime Access Registrar internal subattribute named prepaid.

The IS835C specification requires prepaid users to first be authenticated by the RADIUS server. This requires the configuration of a group service with an authentication service first, followed by the prepaid service that adds prepaid attributes as shown in Setting Up an Authentication Group Service, page 16-5. The group service configuration enables the AA service to add the prepaid subattribute to the response dictionary upon successful authentication, before the prepaid service is invoked.
Configuring IS835C Prepaid Billing

To configure an IS835C prepaid billing service, use the following sections to configure the required Prime Access Registrar objects:

- Setting Up a Prepaid Billing RemoteServer
- Setting Up an IS835C Prepaid Service
- Setting Up Local Authentication
- Setting Up an Authentication Group Service

Setting Up a Prepaid Billing RemoteServer

Prime Access Registrar loads the library dynamically and registers the API functions, then calls out the library initialization API once at startup. The call to initialize functions initializes various data structures and connections with the billing server, as required.

Table 16-1 lists and describes the properties required for an IS835C RemoteServer object.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filename</td>
<td>Name of the shared library provided by the billing server vendor, such as libprepaid.so</td>
</tr>
<tr>
<td>IPAddress</td>
<td>IP address of the billing server</td>
</tr>
<tr>
<td>Port</td>
<td>Port used on the billing server, such as port 66</td>
</tr>
<tr>
<td>Connections</td>
<td>Number of threads the prepaid service and billing server can each use (default is 8).</td>
</tr>
</tbody>
</table>

Setting Up a Prepaid Billing Remote Server

To set up a prepaid billing remote server:

**Step 1**

Use aregcmd to add a RemoteServer under /Radius/RemoteServers.

```
  cd /radius/remoteserver
  add prepaid-is835c
```

**Step 2**

Set remoteserver protocol to prepaid-is835c.

```
  cd prepaid-is835c
  set protocol prepaid-is835c
```

The following is the default configuration of a prepaid-is835c RemoteServer.

```
[ //localhost/Radius/RemoteServers/prepaid-is835c ]
  Name = prepaid-is835c
  Description =
  Protocol =
  IPAddress =
```
Setting Up an IS835C Prepaid Service

Prime Access Registrar uses a service type `prepaid` to support the prepaid billing solution. The prepaid service mediates between the client NAS and the external prepaid billing server.

**Setting Up an IS835C Prepaid Service**
To set up an IS835C prepaid service:

**Step 1** Use `aregcmd` to add a prepaid service under `/Radius/Services`:

```
cd /radius/services
add prepaid
```

```
Added prepaid
```

**Step 2** Set the service type to prepaid.

```
cd prepaid
set type prepaid
```

```
Set Type prepaid
```

A prepaid service has the following default properties:

```
[ //localhost/Radius/Services/prepaid ]
Name = prepaid
Description =
Type = prepaid
IncomingScript- =
OutgoingScript- =
OutagePolicy- = RejectAll
OutageScript- =
MultipleServersPolicy = Failover
RemoteServers/
```

**Step 3** Add a reference to the is835c-prepaid RemoteServer.

```
cd RemoteServer
add 1 prepaid-is835c
```

```
Added 1
```
Setting Up Local Authentication

If you use the Prime Access Registrar server for authentication and authorization in your prepaid billing solution, you should configure an AA service. For example, you might configure a service similar to `local-users` (in the example configuration) for authentication and authorization of local users.

If some of the users are non-prepaid users or if the prepaid users need to have RADIUS authorization attributes returned, you should configure an AA service to perform that authentication and authorization.

**Setting Up a Local Authentication**

To set up a local authentication:

**Step 1** Use `aregcmd` to set up a local authentication service.

```
cd /radius/services
add Prepaid-LocalAuthentication
Added prepaid-LocalAuthentication
```

```
cd prepaid-LocalAuthentication
```

```
[ //localhost/Radius/Services/prepaid-LocalAuthentication ]
  Name = prepaid-LocalAuthentication
  Description =
  Type =
```

**Step 2** Set the service type to local.

```
set type local
Set Type local
```

**Step 3** Set the UserList property to the userlist that contains IS835C prepaid users.

```
set UserList userlist_name
Set UserList userlist_name
```

**Note** You can use an LDAP or ODBC service in place of the local authentication service.

The authentication service must add the Prime Access Registrar internal attribute `prepaid` (subattribute 22) to the response upon successful authentication.

---

Setting Up an Authentication Group Service

Your prepaid billing solution usually requires a group service to tie together an AA service with a prepaid service, a group service to tie together an accounting service with a prepaid service, or both.

If you are using an AA service with your prepaid billing solution, you must configure a group service, for example `prepaid-users`, that ties the requests to the AA service (`local-users` in our example) with the prepaid service.
If you are using Prime Access Registrar for an accounting service with your prepaid billing solution, you must configure a group service, for example `prepaid-file`, that ties accounting requests to both the regular accounting service (`local-file` in our example) and the prepaid service.

**Setting Up an Authentication Group Service**

To set up an authentication group service:

---

**Step 1** Use `aregcmd` to add a prepaid authentication group service under `/Radius/Services`.

```
cd /radius/services
add prepaid-groupAuthentication
Added prepaid-groupAuthentication

cd prepaid-groupAuthentication
[ //localhost/Radius/Services/prepaid-groupAuthentication ]
Name = group-prepaidAuthentication
Description =
Type =
```

**Step 2** Set the service type to group.

```
set type group
Set Type group
```

The group service requires the `ResultRule` to be set to `AND`, the default setting for a group service.

```
is
[ //localhost/Radius/Services/group-prepaidAuthentication ]
Name = group-prepaidAuthentication
Description =
Type = group
IncomingScript- =
OutgoingScript- =
ResultRule = AND
GroupServices/
```

**Step 3** Change directory to `GroupServices` and add references to the prepaid service and the authentication service.

```
cd GroupServices
[ //localhost/Radius/Services/group-prepaidAuthentication/GroupServices ]
add 1 Prepaid-LocalAuthentication
Added 1

add 2 prepaid
Added 2
```
Cisco Real-Time Billing (CRB) is a Cisco proprietary method of providing prepaid billing service. Prime Access Registrar uses vendor-specific attributes (VSA) to extend the standard RADIUS protocol to carry information not usually present in the standard RADIUS packet. Prime Access Registrar uses a set of VSAs allocated to the Cisco VSA pool [26,9].

Prime Access Registrar required several different types of measurements to support a prepaid billing solution. These measurements require the use of metering variables to perform usage accounting. Table 16-2 lists the different measurements and what the AAA client, Prime Access Registrar server, and billing server do with them.

<table>
<thead>
<tr>
<th>Table 16-2 Measurements and Component Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Measurement Type</strong></td>
</tr>
<tr>
<td>Duration</td>
</tr>
<tr>
<td>Total volume</td>
</tr>
<tr>
<td>Uplink volume</td>
</tr>
<tr>
<td>Downlink volume</td>
</tr>
<tr>
<td>Total packets</td>
</tr>
<tr>
<td>Uplink packets</td>
</tr>
</tbody>
</table>
Cisco Prime Access Registrar provides maximum flexibility to billing servers by allowing the metering variable to be modified as the service is used. This requires network nodes to measure all parameters all the time, but to report values only after receiving a reauthorization request.

**Note**

If you have been using an earlier implementation of CRB prepaid billing (Cisco Access Registrar 3.5.2 or earlier), you must recompile the API implementation with the newer API due to the addition of the parameter ebs_context as the first parameter to all API methods. Contact your Cisco systems engineer for assistance with the new API.

This section contains the following topics:

- Configuring CRB Prepaid Billing
- Configuring CRB Prepaid Billing for SSG
- Generic Call Flow
- Vendor-Specific Attributes

**Configuring CRB Prepaid Billing**

To configure an CRB prepaid billing service, use the following sections to configure the required Prime Access Registrar objects:

- Setting Up a Prepaid Billing RemoteServer
- Setting Up a CRB Prepaid Service
- Setting Up a Local Accounting Service
- Setting Up a Local Authentication Service
- Setting Up a Prepaid Accounting Group Service
- Setting Up an Authentication Group Service

If you are using CRB prepaid billing with Service Selection Gateway (SSG), you must also configure extension point scripts and prepaid clients. See Configuring CRB Prepaid Billing for SSG, page 16-15.

**Setting Up a Prepaid Billing RemoteServer**

Table 16-3 lists and describes the properties required for an CRB RemoteServer object.
Chapter 16  Using Prepaid Billing

CRB Prepaid Billing

Setting Up a Prepaid Billing Remote Server

To set up a prepaid billing remote server:

**Step 1**
Use `aregcmd` to add a RemoteServer under `/Radius/RemoteServers`.

```
  cd /radius/remoteservers
  add prepaid-crb
  Added prepaid-crb
```

**Step 2**
Set the RemoteServer protocol to prepaid-crb.

```
  cd prepaid-crb
  set protocol prepaid-crb
  Set Protocol prepaid-crb
```

The following is the default configuration of a prepaid-crb RemoteServer.

```
[ //localhost/Radius/RemoteServers/prepaid-crb ]
  Name = prepaid-crb
  Description =
  Protocol =
  IPAddress =
  Port = 0
  Filename =
  Connections = 8
```

Setting Up a CRB Prepaid Service

Prime Access Registrar uses a service type `prepaid` to support the prepaid billing solution. The prepaid service mediates between the client NAS and the external prepaid billing server.

The prepaid service must receive accounting requests to accurately charge the prepaid billing user. You can also set the prepaid service in a group service to log accounting requests locally or to proxy the accounting requests to another service or to both locations.

### Table 16-3 Prepaid-CRB RemoteServer Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filename</td>
<td>Name of the shared library provided by the billing server vendor, such as <code>libprepaid.so</code></td>
</tr>
<tr>
<td>IPAddress</td>
<td>IP address of the billing server</td>
</tr>
<tr>
<td>Port</td>
<td>Port used on the billing server, such as port 66</td>
</tr>
<tr>
<td>Connections</td>
<td>Number of threads the prepaid service and billing server can each use (default is 8).</td>
</tr>
</tbody>
</table>
**Setting Up a CRB Prepaid Service**

To set up a CRB prepaid service:

---

**Step 1** Use `aregcmd` to add a prepaid service under `/Radius/Services`:

```
cd /radius/services
add prepaid
```

**Step 2** Set the service type to prepaid.

```
cd prepaid
set type prepaid
```

A prepaid service has the following default properties:

```
[ //localhost/Radius/Services/prepaid ]
  Name = prepaid
  Description =
  Type = prepaid
  IncomingScript~ =
  OutgoingScript~ =
  OutagePolicy~ = RejectAll
  OutageScript~ =
  MultipleServersPolicy = Failover
  RemoteServers/
```

**Step 3** Add a reference to the prepaid-crb RemoteServer.

```
cd RemoteServers
add 1 prepaid-crb
```

**Note** The following steps are required only when using Prepaid-CRB with SSG.

**Step 4** Set the IncomingScript to `IncomingScript PPI-Parse-Prepaid-Incoming`.

```
set IncomingScript PPI-Parse-Prepaid-Incoming
```

**Step 5** Set the OutgoingScript to `OutgoingScript PPO-Parse-Prepaid-Outgoing`.

```
set OutgoingScript PPO-Parse-Prepaid-Outgoing
```
Setting Up a Local Accounting Service

If you want to use the Prime Access Registrar server to record the accounting records locally or to forward the accounting records to another RADIUS server, you must configure an accounting service. You might configure a service similar to `local-file` (in the example configuration) for accounting requests. Accounting requests can be logged locally (with an accounting service) or remotely (with a RADIUS service).

If you use the prepaid billing server to generate the accounting records, an accounting service is not necessary.

**Setting Up a Local Accounting Service**

To set up a local accounting service:

**Step 1**

Use `aregcmd` to add a local accounting service under `/Radius/Services`.

```plaintext
cd /radius/services
add prepaid-LocalFileAccounting
```

**Step 2**

Set the service type to file.

```plaintext
cd prepaid-LocalFileAccounting
set type file
```

The file type service has the following properties:

```plaintext
[ //localhost/Radius/Services/prepaid-LocalFileAccounting ]
Name = prepaid-LocalFileAccounting
Description =
Type = file
IncomingScript~ =
OutgoingScript~ =
OutagePolicy~ = RejectAll
OutageScript~ =
FilenamePrefix = accounting
MaxFileSize = "10 Megabytes"
MaxFileAge = "1 Day"
RolloverSchedule =
UseLocalTimeZone = FALSE
```

**Step 3**

Set the FilenamePrefix to Prepaid-Accounting.

```plaintext
set FilenamePrefix Prepaid-Accounting
```

**Step 4**

Set the MaxFileAge to one hour.

```plaintext
set MaxFileAge "1 Hour"
```

The MaxFileSize should remain at the default value of 10 megabytes.
Step 5 Set UseLocalTimeZone to TRUE.

```markdown
set UseLocalTimeZone TRUE
```

Setting Up a Local Authentication Service

If you use the Prime Access Registrar server for authentication and authorization in your prepaid billing solution, you should configure an AA service. For example, you might configure a service similar to `local-users` (in the example configuration) for authentication and authorization of local users.

If some of the users are non-prepaid users or if the prepaid users need to have RADIUS authorization attributes returned, you should configure an AA service to perform that authentication and authorization.

If all of the users in a realm are prepaid users and the prepaid billing client does not require normal RADIUS authorization attributes, an AA service is not necessary.

Setting Up a Local Authentication Service

To set up a local authentication service:

**Step 1** Use `aregcmd` to set up a local authentication service.

```bash
  cd /radius/services
  add Prepaid-LocalAuthentication
  Added prepaid-LocalAuthentication

  cd prepaid-LocalAuthentication
  [ //localhost/Radius/Services/prepaid-LocalAuthentication ]
  Name = prepaid-LocalAuthentication
  Description =
  Type =
```

**Step 2** Set the service type to local.

```bash
  set type local
  Set Type local
```

**Step 3** Set the UserList property to the userlist that contains IS835C prepaid users.

```bash
  set UserList userlist_name
  Set UserList userlist_name
```

Note You can use an LDAP or ODBC service in place of the local authentication service.
Setting Up a Prepaid Accounting Group Service

A prepaid billing solution usually requires a group service to tie together an AA service with a prepaid service, a group service to tie together an accounting service with a prepaid service, or both.

If you are using an AA service with your prepaid billing solution, you must configure a group service, for example prepaid-users, that ties the requests to the AA service (local-users in our example) with the prepaid service.

If you are using Prime Access Registrar for an accounting service with your prepaid billing solution, you must configure a group service, for example prepaid-file, that ties accounting requests to both the regular accounting service (local-file in our example) and the prepaid service.

Setting Up a Prepaid Accounting Group Service

To set up a prepaid accounting group service:

Step 1 Use aregcmd to create an accounting group service under /Radius/Services.

```
cd /radius/services
add Prepaid-Accounting
Added prepaid-accounting
```

Step 2 Set the service type to group.

```
cd prepaid-accounting
[ //localhost/Radius/Services/prepaid-accounting ]
Name = prepaid-accounting
Description =
Type =

set type group

Set Type group
```

The group service has the following properties:

```
[ //localhost/Radius/Services/prepaid-accounting ]
Name = prepaid-accounting
Description =
Type = group
IncomingScript- =
OutgoingScript- =
ResultRule = AND
GroupServices/
```

Step 3 Reference the Prepaid and Prepaid-LocalAccounting services under GroupServices.

```
cd GroupServices
[ //localhost/Radius/Services/prepaid-accounting/GroupServices ]

add 1 prepaid

Added 1
```
Setting Up an Authentication Group Service

A prepaid billing solution usually requires a group service to tie together an AA service with a prepaid service, a group service to tie together an accounting service with a prepaid service, or both.

If you are using an AA service with your prepaid billing solution, you must configure a group service, for example prepaid-users, that ties the requests to the AA service with the prepaid service.

If you are using Prime Access Registrar for an accounting service with your prepaid billing solution, you must configure a group service, for example prepaid-file, that ties accounting requests to both the regular accounting service and the prepaid service.

Setting Up an Authentication Group Service

To set up an authentication group service:

**Step 1**  Use aregcmd to add a prepaid authentication group service under /Radius/Services.

```bash
add 2 prepaid-LocalFileAccounting
```

```
Added 2
```

**Step 2**  Set the service type to group.

```bash
set type group
```

Set Type group

The group service requires the ResultRule to be set to AND, the default setting for a group service.

```
ls
```

```
[ //localhost/Radius/Services/group-prepaidAuthentication ]
Name = group-prepaidAuthentication
Description =
Type =
IncomingScript~ =
OutgoingScript~ =
ResultRule = AND
GroupServices/
```
**Step 3** Change directory to GroupServices and add references to the prepaid service and the authentication service.

```bash
cd GroupServices
[ //localhost/Radius/Services/group-prepaidAuthentication/GroupServices ]

add 1 Prepaid-LocalAuthentication
Added 1

add 2 prepaid
Added 2
```

### Configuring CRB Prepaid Billing for SSG

In addition to the configuration described in **CRB Prepaid Billing, page 16-7**, when using CRB-Prepaid billing with SSG, you must also perform the following:

- Setting Up an Outgoing Script
- Setting Up an Incoming Script
- Setting Up a Prepaid Outgoing Script
- Adding Prepaid Clients

#### Setting Up an Outgoing Script

To set up an outgoing script:

**Step 1** Use `aregcmd` to add the `PCO-Parse-Client-Outgoing` outgoing script under `/Radius/Scripts`:

```bash
cd /radius/scripts
add PCO-Parse-Client-Outgoing
Added PCO-Parse-Client-Outgoing

cd PCO-Parse-Client-Outgoing
[ //localhost/Radius/Scripts/PCO-Parse-Client-Outgoing ]
Name = PCO-Parse-Client-Outgoing
Description =
Language =
```

**Step 2** Set the language to tcl.

```bash
set language tcl

Set Language tcl
```

**Step 3** Set the filename to `PCO-parse.client-outgoing.tcl`.

```bash
set filename PCO-parse.client-outgoing.tcl
```
Set Filename PCO-parse.client-outgoing.tcl

**Step 4** Set the EntryPoint to PCO-parse-client-outgoing.

```tcl
set EntryPoint PCO-parse-client-outgoing
```

Setting Up an Incoming Script

To set up an incoming script:

**Step 1** Use `aregcmd` to add the `PPI-Parse-Prepaid-Incoming` script under `/Radius/Scripts`.

```bash
cd /radius/scripts
add PPI-Parse-Prepaid-Incoming
```

**Step 2** Set the language to `tcl`.

```bash
cd PPI-Parse-Prepaid-Incoming
set language tcl
```

**Step 3** Set the filename to `PPI-Parse-Prepaid-Incoming.tcl`.

```tcl
set filename PPI-Parse-Prepaid-Incoming.tcl
```

**Step 4** Set the EntryPoint to `PPO-Parse-Prepaid-Outgoing`.

```tcl
set EntryPoint PPO-Parse-Prepaid-Outgoing
```

Setting Up a Prepaid Outgoing Script

To set up a prepaid outgoing script:

**Step 1** Use `aregcmd` to add the `PPO-Parse-Prepaid-Outgoing` outgoing script under `/Radius/Scripts`:

```bash
cd /radius/scripts
```

**Step 2** Add the `PPO-Parse-Prepaid-Outgoing` outgoing script under `/Radius/Scripts`.

```bash
cd /radius/scripts
add PPO-Parse-Prepaid-Outgoing
```

Added `PPO-Parse-Prepaid-Outgoing`
Step 3 Set the language to tcl.

```bash
cd PPO-Parse-Prepaid-Outgoing
set language tcl
```

Step 4 Set the filename to `PPO-Parse-Prepaid-Outgoing.tcl`.

```bash
set filename PPO-Parse-Prepaid-Outgoing.tcl
```

Step 5 Set the EntryPoint to `PPO-Parse-Prepaid-Outgoing`.

```bash
set EntryPoint PPO-Parse-Prepaid-Outgoing
```

---

**Adding Prepaid Clients**

To add prepaid clients:

Step 1 Use `aregcmd` to add the prepaid clients under `/Radius/Clients`.

```bash
cd /radius/clients
add SSG
```

A RADIUS client has the following properties:

```bash
[ //localhost/Radius/Clients/ssg ]
Name = ssg
Description = 
IPAddress =
SharedSecret =
Type = NAS
Vendor =
IncomingScript~ =
OutgoingScript~ =
EnableDynamicAuthorization = FALSE
NetMask =
```

Step 2 Set the IPAddress property to the client IP address.

```bash
set IPAddress aaa.bbb.ccc.ddd
```

Step 3 Set the SharedSecret.

```bash
set sharedsecret cisco
```

Step 4 Set the OutgoingScript to `PCO-Parse-Client-Outgoing`.

```bash
set OutgoingScript PCO-Parse-Client-Outgoing
```
set out PCO-Parse-Client-Outgoing

Set OutgoingScript PCO-Parse-Client-Outgoing

Generic Call Flow

This section describes the generic call flow for the Prime Access Registrar CRB prepaid billing. The call flow is controlled by the AAA client. The Prime Access Registrar server converts VSAs into calls to the billing server. For information about call flows and attributes for IS835C, see IS835C Prepaid Billing, page 16-2.

The packet flows presented in Figure 16-1 are specific to the Prime Access Registrar CRB prepaid billing only. The headlines in the packet flows are general and do represent all data transferred. The letters c, s, and b in Figure 16-1 designate the packet’s source of client, server, or billing server, respectively.

Figure 16-1  Generic Call Flow Diagram

<table>
<thead>
<tr>
<th>AAA client</th>
<th>AAA server</th>
<th>Billing server</th>
</tr>
</thead>
<tbody>
<tr>
<td>1c</td>
<td>Access-Request</td>
<td>1s aaa_ebs_init_authenticate()</td>
</tr>
<tr>
<td></td>
<td>CRB_AR_INIT_AUTHENTICATE</td>
<td></td>
</tr>
<tr>
<td>2c</td>
<td>Access-Accept</td>
<td>2s authenticate result</td>
</tr>
<tr>
<td>3c</td>
<td>Access-Request</td>
<td>3s aaa_ebs_init_authorize()</td>
</tr>
<tr>
<td></td>
<td>CRB_AR_INIT_AUTHORIZE</td>
<td></td>
</tr>
<tr>
<td>4c</td>
<td>Access-Request+VSAs</td>
<td>4s return quota array</td>
</tr>
<tr>
<td>5c</td>
<td>Accounting-Start</td>
<td>6s</td>
</tr>
<tr>
<td>6c</td>
<td>Accounting-Resp</td>
<td></td>
</tr>
<tr>
<td>7c</td>
<td>Access-Request+VSAs</td>
<td>7s aaa_ebs_reauthorize()</td>
</tr>
<tr>
<td>8c</td>
<td>Access-Accept+VSAs</td>
<td>8s Updated quota</td>
</tr>
<tr>
<td>9c</td>
<td>Accounting-Stop+VSAs</td>
<td>9s aaa_ebs_return_quota()</td>
</tr>
<tr>
<td>10c</td>
<td>Accounting-Resp</td>
<td>10s Final status</td>
</tr>
</tbody>
</table>

Data Flow
This section contains the following topics:

- Access-Request (Authentication)
- Access-Accept (Authentication)
- Access-Request (Authorization)
- Access-Accept (Authorization)
- Accounting-Start
- Data Flow
- Access-Request (Quota Depleted)
- Accept-Accept (Quota Depleted)
- Accounting Stop (Session End)
- Accounting Response (Final Status)

**Access-Request (Authentication)**

_Flow 1c_ shows the client sending the Access-Request to AAA Server, part of a normal authentication request. The exact nature of the message contents is dictated by the access technology, be it be CDMA1X-RTT, GPRS, or another. The Access-Request might involve other messages such as PAP/CHAP or another form of authentication.

The _Flow 1c_ Access-Request might contain a prepaid specific VSA, CRB_AUTH_REASON. Table 16-4 lists the attributes included in the authentication Access-Request. This tells the Prime Access Registrar server to authenticate the subscriber with the Prepaid server as well. If the value is CRB_AR_INIT_AUTHENTICATE, the initial quota must be obtained for a single service prepaid solution. If this VSA is not present, the Prime Access Registrar server will not authenticate with the Prepaid billing server.

**Table 16-4 Attributes Sent During Subscriber Authentication**

<table>
<thead>
<tr>
<th>Attribute Number</th>
<th>Attribute Name</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User-Name</td>
<td>APPL: Mobile Node Username</td>
<td>Required</td>
</tr>
<tr>
<td>2</td>
<td>NAS IP Address</td>
<td>Accounting Node IP Address</td>
<td>APPL: Required, POA</td>
</tr>
<tr>
<td>31</td>
<td>Calling-station-ID</td>
<td>APPL:MSISDN or IMSI</td>
<td>APPL: Conditional</td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_AUTH_REASON</td>
<td>See VSA section</td>
<td>Required</td>
</tr>
<tr>
<td></td>
<td>CRB_AR_INIT_AUTHENTICATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_USER_ID</td>
<td>APPL:PDSN address or SSG address</td>
<td>APPL: Required, Address of the PDSN</td>
</tr>
</tbody>
</table>
In **Flow 1s**, the Prime Access Registrar server sends a call to the billing server to authenticate the prepaid user and possibly determine more information about the subscriber’s account. The Prime Access Registrar server can be configured to generate this packet flow, using a subscriber profile parameter, if the request is from a prepaid subscriber.

### Access-Accept (Authentication)

**Flow 2b** shows the billing server returning the authentication result. The billing server returns a failure if the prepaid subscriber has an inadequate balance.

**Flow 2s** shows the Prime Access Registrar server sending the Access-Accept to the AAA client. This message flow contains at least one prepaid billing-specific VSA (listed in Table 16-5) and might contain other access technology-specific attributes.

#### Table 16-4 Attributes Sent During Subscriber Authentication (continued)

<table>
<thead>
<tr>
<th>Attribute Number</th>
<th>Attribute Name</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>26, 9</td>
<td>CRB_SERVICE_ID</td>
<td>APPL: Service ID such as Simple IP service, Mobile IP service, or VPN service</td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_SESSION_ID</td>
<td>This VSA contains the session key ID information</td>
<td>Required; the session ID must be globally unique across all clients and across reboots of the client</td>
</tr>
</tbody>
</table>

#### Access-Request (Authorization)

In **Flow 3c**, the AAA client sends another Access-Request, this time to authorize the subscriber. **Table 16-6** lists the attributes required by the Prime Access Registrar server to authorize the subscriber. The session key ID used must be specified using a prepaid VSA pointing to the RADIUS attribute (standard or VSA).

#### Table 16-5 Attributes Sent to AAA client in Access-Accept (Authentication)

<table>
<thead>
<tr>
<th>Attribute Number</th>
<th>Attribute Name</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>26, 9</td>
<td>CRB__USER_TYPE</td>
<td>See <strong>Vendor-Specific Attributes, page 16-25</strong></td>
<td>Optional</td>
</tr>
<tr>
<td></td>
<td>CRB_AR_INIT_AUTHENTICATE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Table 16-6 Attributes Sent During Subscriber Authorization

<table>
<thead>
<tr>
<th>Attribute Number</th>
<th>Attribute Name</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User-Name</td>
<td>APPL: Mobile Node Username</td>
<td>Required</td>
</tr>
<tr>
<td>2</td>
<td>NAS IP Address</td>
<td>Accounting Node IP Address</td>
<td>APPL: Required, POA</td>
</tr>
</tbody>
</table>
In Flow 3s, the Prime Access Registrar server sends the Prepaid billing server to obtain a quota. The quota might contain several values depending on the number of measurement parameters chosen.

### Access-Accept (Authorization)

**Flow 4b** shows the billing server returning the quota array for the subscriber.

In Flow 4s, the Prime Access Registrar server converts the quota array received into VSAs and sends an Access-Accept with the assembled VSAs to the AAA client. Table 16-7 lists the prepaid-specific VSAs that might be included in the Access-Accept response message sent to the AAA client. For more detailed information about the VSAs, see Vendor-Specific Attributes, page 16-25.

<table>
<thead>
<tr>
<th>Attribute Number</th>
<th>Attribute Name</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>26, 9</td>
<td>CRB_AUTH_REASON CRB_AR_INIT_AUTHORIZE</td>
<td>See Vendor-Specific Attributes, page 16-25</td>
<td>Required</td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_USER_ID</td>
<td>APPL: PDSN address or SSG address</td>
<td>APPL: Required, Address of the PDSN</td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_SERVICE_ID</td>
<td>APPL: Service ID such as Simple IP service, Mobile IP service, or VPN service</td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_SESSION_ID</td>
<td>This VSA contains the session key ID information</td>
<td>Required; the session ID must be globally unique across all clients and across reboots of the client</td>
</tr>
</tbody>
</table>

Flows 3c through 4s are repeated for every service started or restarted by the AAA client.

However, if the return parameters indicate that the authorization is rejected, an Access-Accept message is generated and sent to the client as shown in Table 16-8. When this type of error condition occurs, no other VSA is included in the Access-Accept message.


**Accounting-Start**

In **Flow 5c**, the AAA client sends the Accounting-Start. In **Flow 6s**, the Prime Access Registrar server replies with the Accounting-Response.

**Data Flow**

At this point, the data transfer begins. The AAA client monitors the subscriber’s allocated quotas for metering parameters. A subscriber’s Reauthorization request is generated when a quota for at least one of the metering parameters, is depleted.

**Access-Request (Quota Depleted)**

**Flow 7c** shows the client sending an Access-Request to the Prime Access Registrar server because at least one quota has been depleted. The Access-Request includes different measurements of how much of the quotas were used in VSA format. This enables the billing server to account for the usage and manage the subscriber’s balance before assigning a new quota. **Table 16-9** lists the attributes returned to the Prime Access Registrar server:

<table>
<thead>
<tr>
<th>Attribute Number</th>
<th>Attribute Name</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User-Name</td>
<td>APPL: Mobile Node Username</td>
<td>Conditional</td>
</tr>
<tr>
<td>2</td>
<td>NAS IP Address</td>
<td>Accounting Node IP Address</td>
<td>APPL: Required, POA address, or Home Node address</td>
</tr>
<tr>
<td>31</td>
<td>Calling-station-ID</td>
<td>APPL: MSISDN or IMSI</td>
<td>APPL: Conditional</td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_AUTH_REASON</td>
<td>See VSA</td>
<td>Required</td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_USER_ID</td>
<td>APPL: PDSN address or SSG address</td>
<td>APPL: Required, address of SGSN</td>
</tr>
</tbody>
</table>
Accept-Accept (Quota Depleted)

Flow 7s shows the Prime Access Registrar server returning the used quota array to the billing server. The call includes `aaa_ebs_reauthorize()`. The billing server sends an updated quota array for the next period to the Prime Access Registrar server.

In Flow 8s, the Prime Access Registrar server converts the quota array into VSAs and sends them to the AAA client.

<table>
<thead>
<tr>
<th>Attribute Number</th>
<th>Attribute Name</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>26, 9</td>
<td>CRB_DURATION</td>
<td>See Vendor-Specific Attributes, page 16-25</td>
<td>Required</td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_TOTAL_VOLUME</td>
<td></td>
<td>Conditional</td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_UPLINK_VOLUME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_DOWNLINK_VOLUME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_TOTAL_PACKETS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_UPLINK_PACKETS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_DOWNLINK_PACKETS</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Accounting Stop (Session End)

In Flow 9c, the client sends an Accounting-Stop to the Prime Access Registrar server to end the session. The Accounting-Stop message includes an updated quota array with the usage adjustments since the previous authorization in the VSA form.

Table 16-11 lists the attributes included in the Accounting-Stop message set to the Prime Access Registrar server and forwarded to the billing server.

Accounting Response (Final Status)

In Flow 9s, the Prime Access Registrar server sends the used quota array to the billing server in an Accounting-Stop message. Any values returned by the billing server in Flow 10b are discarded.
Flow 10s shows the Prime Access Registrar server sending final Accounting-Response message to the AAA client.

Table 16-11  Attributes Sent in Accounting-Stop Message

<table>
<thead>
<tr>
<th>Attribute Number</th>
<th>Attribute Name</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User-Name</td>
<td>APPL: Mobile Node Username</td>
<td>Conditional</td>
</tr>
<tr>
<td>2</td>
<td>NAS IP Address</td>
<td>Accounting Node IP Address</td>
<td>APPL: Required, POA</td>
</tr>
<tr>
<td>31</td>
<td>Calling-station-ID</td>
<td>APPL: MSISDN or IMSI</td>
<td>APPL: Conditional</td>
</tr>
<tr>
<td>40, 2</td>
<td>Acct_status_type</td>
<td>Indicates the accounting “Stop” for the service</td>
<td>Required; this value (2) indicates an Accounting-Stop request message</td>
</tr>
<tr>
<td>42</td>
<td>Acct-Input-Octets</td>
<td>The number of octets sent by the subscriber; uplink</td>
<td>Required</td>
</tr>
<tr>
<td>43</td>
<td>Acc_Output_Octets</td>
<td>The number of octets received by the subscriber; downlink</td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>Acct-Session-Time</td>
<td>Duration of the session</td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>Acct-Input-Packets</td>
<td>Number of packets sent by the subscriber</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>Acct-Output-Packets</td>
<td>Number of packets received by the subscriber</td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>Acct-Terminate-Cause</td>
<td>This parameter, used for tracking, should remain the same for all accounting requests for a given service.</td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_DURATION</td>
<td>See Vendor-Specific Attributes, page 16-25</td>
<td>Conditional</td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_TOTAL VOLUME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_UPLINK VOLUME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_DOWNLINK VOLUME</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_TOTAL_PACKETS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_UPLINK_PACKETS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_DOWNLINK_PACKETS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26, 9</td>
<td>CRB_SESSION_ID</td>
<td>Specifies the RADIUS attribute carrying the session ID information</td>
<td>Optional</td>
</tr>
</tbody>
</table>
Vendor-Specific Attributes

Vendor-specific attributes are included in specific RADIUS packets to communicate prepaid user balance information from the Prime Access Registrar server to the AAA client, and actual usage, either interim or total, between the NAS and the Prime Access Registrar Server.

Table 16-12 lists the VSAs that will be defined in the API. Table 16-12 also lists the string to be used with Cisco-AVPair below the VSA.

VSAs that start with CRB are used for Cisco Radius Billing prepaid service.

Table 16-12  Vendor-Specific Attributes for the Cisco Prepaid Billing Solution

<table>
<thead>
<tr>
<th>VSA Name</th>
<th>Type</th>
<th>Source (Call Flow)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRB_AUTH_REASON</td>
<td>Int8</td>
<td>1c, 7c, 7c</td>
<td>Passed with re-authorization: 1. Initial Authentication</td>
</tr>
<tr>
<td>crb-auth-reason</td>
<td></td>
<td></td>
<td>2. Initial Authorization</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3. Re-authorization</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4. Return Quota</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5. Query to EBS</td>
</tr>
<tr>
<td>CRB_USER_ID</td>
<td>String</td>
<td>1c, 7c, 7c</td>
<td>APPL: In PDSN this can be Address of the PDSN.</td>
</tr>
<tr>
<td>crb-user-id</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRB_SERVICE_ID</td>
<td>String</td>
<td>1c, 7c</td>
<td>Identifies the subscriber's service</td>
</tr>
<tr>
<td>crb-service-id</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRB_USER_TYPE</td>
<td>Int8</td>
<td>4s</td>
<td>Type of user: 1. Prepaid user</td>
</tr>
<tr>
<td>crb-entity-type</td>
<td></td>
<td></td>
<td>2. Post-paid with no credit limit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3. Post-paid with credit limit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4. Invalid user</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The source for this VSA value could be from the Subscriber profile or from</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>the billing server</td>
</tr>
</tbody>
</table>
### Table 16-12  Vendor-Specific Attributes for the Cisco Prepaid Billing Solution (continued)

<table>
<thead>
<tr>
<th>VSA Name</th>
<th>Type</th>
<th>Source (Call Flow)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRB_DURATION</td>
<td>Int32</td>
<td>4s, 8s</td>
<td>Downlink quota received by the AAA client</td>
</tr>
<tr>
<td>crb-duration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRB_TOTAL_VOLUME</td>
<td>Int32</td>
<td></td>
<td>Total Volume quota received by the AAA client</td>
</tr>
<tr>
<td>crb-total-volume</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRB_UPLINK_VOLUME</td>
<td>Int32</td>
<td></td>
<td>Uplink volume quota received by the AAA client</td>
</tr>
<tr>
<td>crb-uplink-volume</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRB_DOWNLINK_VOLUME</td>
<td>Int32</td>
<td></td>
<td>Uplink Volume quota received by the AAA client</td>
</tr>
<tr>
<td>crb-downlink-volume</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRB_TOTAL_PACKETS</td>
<td>Int32</td>
<td></td>
<td>Downlink Packet quota received by the AAA client</td>
</tr>
<tr>
<td>crb-total-packets</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRB_UPLINK_PACKETS</td>
<td>Int32</td>
<td></td>
<td>Uplink Packet quota received by the AAA client</td>
</tr>
<tr>
<td>crb-uplink-packets</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRB_DOWNLINK_PACKETS</td>
<td>Int32</td>
<td></td>
<td>Uplink Volume quota received by the AAA client</td>
</tr>
<tr>
<td>crb-downlink-packets</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CRB_SESSION_ID</td>
<td>String</td>
<td></td>
<td>Additional field if session ID is required. This VSA provides the real time billing-specific session ID. This VSA duplicates the contents of the technology-specific session ID or the contents of RADIUS attributes 44 or 50. The NAS can use this VSA to generate a unique session ID. If this VSA is not present, then RADIUS attribute 44 is used instead. If this is a string AV Pair-type attribute, the name is the string attribute name.</td>
</tr>
<tr>
<td>crb-session-id</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Implementing the Prepaid Billing API

A shared library must implement the API functions to perform the various tasks given in the description of each of the function. This needs to be compiled as a shared library and then specified as part of the remote server configuration at the Filename property. See Setting Up a Prepaid Billing RemoteServer, page 16-3 or Setting Up a Prepaid Billing RemoteServer, page 16-8.

At startup, Prime Access Registrar loads the library dynamically and registers the API functions, then calls out the appropriate API functions present in the shared library. The values for the arguments passed to these API calls are purely derived from the incoming RADIUS packet and Prime Access Registrar does not maintain any dynamic information related to the call flow. It is up to the API function to make use of the information passed to it as C structures to contact the Billing server, get appropriate data, and return the same to Prime Access Registrar using the designated arguments.

Note Cisco works with you to develop the prepaid billing service and implement the API. For more information, contact your Cisco systems engineer.

At various times, according to the call flow described in the Prepaid Call Flow Specification (CRB or IS835C), Prime Access Registrar calls out appropriate API functions present in the shared library. The values for the arguments passed to these API calls are purely derived from the incoming RADIUS packet and Prime Access Registrar does not maintain any dynamic information related to the call flow. It is up to the API function to make use of the information passed to it as C structures to contact the Billing server, get appropriate data, and return the same to Prime Access Registrar using the designated arguments.

Note See the API specifications for more details pertaining to the arguments and return values of the API.
Using Cisco Prime Access Registrar Server Features

This chapter provides information about how to use the Cisco Prime Access Registrar (Prime Access Registrar) server features.

This chapter contains the following sections:

- Incoming Traffic Throttling
- Backing Store Parsing Tool
- Configurable Worker Threads Enhancement
- Session-Key Lookup
- Query-Notify
- Support for Windows Provisioning Service
- Command Completion
- Service Grouping Feature
- SHA-1 Support for LDAP-Based Authentication
- Dynamic Attributes
- Tunneling Support Feature
- xDSL VPI/VCI Support for Cisco 6400
- Apply Profile in Cisco Prime Access Registrar Database to Directory Users
- Directory Multi-Value Attributes Support
- MultiLink-PPP (ML-PPP)
- Dynamic Updates Feature
- NAS Monitor
- Automatic Information Collection (arbug)
- Simultaneous Terminals for Remote Demonstration
- Support for RADIUS Check Item Attributes
- User-Specific Attributes
- Packet of Disconnect
- Dynamic DNS
Incoming Traffic Throttling

Prime Access Registrar offers two options to tackle traffic bursts by limiting incoming traffic. You will find two properties, MaximumIncomingRequestRate and MaximumOutstandingRequests, under `/Radius/Advanced` to limit the incoming traffic.

This contains the following sections:

- MaximumIncomingRequestRate
- MaximumOutstandingRequests

MaximumIncomingRequestRate

You can use the MaximumIncomingRequestRate property to limit incoming traffic in terms of “allowed requests per second”.

For example, if you set the MaximumIncomingRequestRate to $n$, then at any given second, only $n$ requests are accepted for processing. In the next second, another $n$ requests are accepted regardless of whether the requests accepted earlier are processed or not. This condition serves as a soft limit.

The MaximumIncomingRequestRate property by default is zero (disabled).

MaximumOutstandingRequests

You can use the MaximumOutstandingRequests property to limit incoming traffic in terms of “requests processed”.

For example, if you set the MaximumOutstandingRequests to $n$, $n$ requests are accepted for processing. Further requests are accepted only after processing some of these requests and sending the replies back. This condition serves as a hard limit.

The MaximumOutstandingRequests property by default is zero (disabled).

**Note**

You can enable either of these properties independent of the other.

**Configuring the MaximumOutstandingRequests**

To configure the MaximumIncomingRequestRate or MaximumOutstandingRequests property:

**Step 1**  Log into `aregcmd`.

**Step 2**  Change directory to `/Radius/Advanced`. 
Step 3  Set the MaximumIncomingRequestRate or MaximumOutstandingRequests property to non-zero values.

```plaintext
set MaximumIncomingRequestRate n
```
or

```plaintext
set MaximumOutstandingRequests n
```
where \( n \) is any nonzero value.

Step 4  Save the configuration; enter:

```plaintext
save
```

Step 5  Reload the server; enter:

```plaintext
reload
```

---

**Backing Store Parsing Tool**

Prime Access Registrar tool, `carbs.pl`, helps to analyze the session backing store files. You will find this tool under `/cisco-ar/bin` directory.

Using carbs.pl, you can:

- Get information about the active, stopped, and stale RADIUS sessions.
- Clear phantom sessions manually.
- Process the binary log files and get information in a user-readable format.

The syntax is:

```plaintext
```

- `-a`—All session statistics (active, stale, stopped)
- `-d`—<Directory> Default: .
- `-f`—<Filename> Default: 00*.log
- `-v`—verbose Default: off
- `-p`—Clear phantom sessions
- `-o`—<Filename> Output log to TEXT
- `-h`—Help, usage

**Table 17-1** lists the options available with carbs.pl and their description.
### Configurable Worker Threads Enhancement

Prime Access Registrar provides a configurable variable you can use to increase the number of worker threads to handle a greater number of RADIUS packets during peak operating periods. This variable controls the processing of greater number of RADIUS packets than expected during peak operating periods.

The variable, RADIUS_WORKER_THREAD_COUNT, is found in the arserver file under /cisco-ar/bin/arserver and controls the number of worker threads the Prime Access Registrar server creates. You can increase the number of worker threads to help make more efficient use of the server’s CPU.

**Note**

Before you increase the setting for RADIUS_WORKER_THREAD_COUNT, you should be certain that you are running into a worker thread starvation issue. If you use scripts that consume a lot of processing and memory, you might run out of memory if you create too many worker threads.

Increasing the number of worker threads also increases memory utilization.

---

#### Table 17-1  Carbs.pl Options and Description

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d&lt;directory&gt;</td>
<td>Optional. Accepts a directory as parameter with no trailing slash. You can use this option to change the default directory to scan for BackingStore log files. Default is current directory.</td>
</tr>
<tr>
<td>-f&lt;logfile&gt;</td>
<td>Optional. Accepts a logfile as parameter with no leading or trailing slashes. You can use this option to change the default log files. Allows you to enter individual logfile name as well as wildcard characters surrounded by single quotes.</td>
</tr>
<tr>
<td>-v</td>
<td>Optional. No parameters. You can use this option to get total session count and phantom session count.</td>
</tr>
<tr>
<td>-p</td>
<td>Optional. No parameters. Generates a list of phantom sessions. You can use this option to clear the stale sessions.</td>
</tr>
<tr>
<td>-o</td>
<td>Optional. Accepts &lt;output file&gt; as parameter. You can use this option to convert BackingStore log files to readable files and write the results to the output file specified.</td>
</tr>
<tr>
<td>-a</td>
<td>Optional. No parameters. You can use this option to print all session statistics, such as per-NAS stale session count, total active sessions, and total stale sessions.</td>
</tr>
<tr>
<td>-h</td>
<td>You can use this option to get help with usage of carbs.pl.</td>
</tr>
</tbody>
</table>
The purpose of this enhancement is to take advantage of spare CPU bandwidth which was not being used in earlier releases of Prime Access Registrar due to a lower number of worker threads. At times, the worker threads would be stuck doing work that took a long time to complete, like running a script. Having more threads will help mitigate these situations and will help improve on the latency created due to lack of free worker threads.

**Note** Before modifying the RADIUS_WORKER_THREAD_COUNT variable, consult with a TAC representative to ensure that modifying the RADIUS_WORKER_THREAD_COUNT is warranted.

### Modifying the RADIUS WORKER THREAD COUNT

To modify the RADIUS_WORKER_THREAD_COUNT variable:

1. Log into the Prime Access Registrar server as a root user and change directory to `/cisco-ar/bin`.
2. Use a text editor and open the `arserver` file.
3. Locate the line with the RADIUS_WORKER_THREAD_COUNT variable.

   ```
   #change this to configure number of worker threads
   RADIUS_WORKER_THREAD_COUNT=256
   ```

4. Modify the number of RADIUS worker threads to the number you choose.

   **Note** There is no upper limit to the number of RADIUS worker threads you can enable in your Prime Access Registrar server, but you should take care not to exceed your server’s memory capacity.

5. Save the file and restart the Prime Access Registrar server.

---

### Session-Key Lookup

The Session-Key Lookup feature enables you to identify the Session Manager and Session Key of an existing session based on certain attributes associated with that session, such as the Mobile Station Integrated Services Digital Network (MSISDN) number.

The Session-Key Lookup feature requires the following enhancements to Prime Access Registrar software:

- Enabling a query service to be invoked for Ascend-IP-Allocate packets
- Enabling the setting of the Session-Key and Session-Manager environment variables by a query operation
- Performing session management after the query operation
- A new environment variable, Set-Session-Mgr-And-Key-Upon-Lookup, which when set to TRUE causes a session-cache Resource Manager to set the Session-Manager and Session-Key environment variables during the query lookup.

The Session-Key Lookup feature is useful in a scenario where an existing session requires an update from an incoming Ascend-IPA-Allocate packet (from a different NAS or device) with modified authorization attributes. Note that this Ascend-IPA-Packet might not have the exact set of attributes as...
Chapter 17 Using Cisco Prime Access Registrar Server Features

Query-Notify

The original packet that created the session. However, the Ascend-IPA-Allocate packet must contain at least one attribute that can uniquely identify the session (such as the MSISDN number) and should contain the same UserName of the original session.

The Session-Key Lookup feature works in tandem with the RADIUS Query feature, where a RADIUS Query service is defined with the unique attribute (such as the MSISDN number) as the query-key and is configured to query all session managers. The Query-Service environment variable is set to the defined RADIUS Query service and the new environment variable (Set-Session-Mgr-And-Key-Upon-Lookup) is set to TRUE for this Ascend-IPA-Allocate packet. This triggers a query operation on all the live sessions. If there is a match, the Session-Manager and Session-Key of that session is used for subsequent session management. During session management, the session cache is updated with the modified authorization attributes.

The Session-Manager OutgoingScript (or any outgoing script that executes after the Session-Manager Outgoing Script) should not reject the packet when doing a Session-Key lookup. Doing so causes the session to be deleted.

Query-Notify

The Query-Notify feature enables you to store information about Wireless Application Protocol (WAP) gateways that have queried for User Identity-IP Address mapping and send appropriate messages to the WAP gateway when the subscriber logs out of the network.

Prime Access Registrar has been enhanced to update the session cache with the attribute-value pairs of an interim accounting update packet. This ensures the Prime Access Registrar server provides updated or current information to the WAP gateway during the proxy of interim records or query of the session cache.

Prime Access Registrar has been enhanced to also notify the WAP gateways that have queried a session with interim accounting update packets. If a WAP gateway does not respond to the Interim accounting update packets, the Prime Access Registrar server times out and retries by notifying the WAP gateways again. If there is no response after all the retries, the proxy packet is deleted and no change is made to the session or the WAP gateway’s state in the Prime Access Registrar server. You can configure the number of retries under /Radius/Clients/notificationproperties.

The accounting response packet from the Prime Access Registrar server to the GPRS Gateway Support Node (GGSN) is independent of the proxy operation to the WAP gateways. The accounting response packet is sent back immediately without waiting for responses from the WAP gateways.

The Query-Notify feature also enables you to quarantine IP addresses for a configurable amount of time if a WAP gateway does not respond to Accounting-Stop packets sent by the Prime Access Registrar server.

The Prime Access Registrar server stores information about clients (usually the IP address) that queried for particular user information and sends RADIUS Accounting-Stop packets to those clients when the Prime Access Registrar server receives the Accounting-Stop packet. There is no intermediate proxy server between the Prime Access Registrar server and the WAP gateway.

To support the Query-Notify feature, the Prime Access Registrar server’s radius-query service has been modified to also store information like the IP address about the clients queried for cached information. The information is stored in the user session record along with the cached information so it is available after a server reload.
Configuring the Query-Notify feature

To configure the Query-Notify feature:

**Step 1** Configure the Clients object under `/Radius/Clients`.

**Step 2** Set the EnableNotifications property to TRUE.

The EnableNotifications property indicates that a client can receive Accounting-Stop notifications from the Prime Access Registrar server. When EnableNotifications is set to TRUE, a sub-directory named NotificationProperties appears in client object configuration.

**Step 3** Configure the properties under the client’s NotificationProperties subdirectory.

See [Clients, page 4-6](#), for information about how to configure these properties.

**Step 4** Configure a list of attributes to store under `/Radius/Advanced/Attribute Groups/<Notification Group>` where `<notification group>` is the name of an Attribute Group containing a list of attributes to be stored.

This section contains the following topics:

- **Call Flow**
- **Configuration Examples**
- **Memory and Performance Impact**

### Call Flow

This section describes the call flow of the Query-Notify feature.

1. The Prime Access Registrar server caches information from an from Accounting-Start.
   
   This information is usually from a GGSN when a subscriber enters into the network.

2. When a WAP gateway receives a request to authenticate a subscriber, it queries the Prime Access Registrar server using an Access-Request packet to retrieve the cached information for that subscriber.

3. The Prime Access Registrar server responds with Access-Accept if an entry is found for the subscriber in its cache; otherwise the server returns an Access-Reject.

   The Prime Access Registrar server sends an Access-Accept packet to the WAP gateway. The list of attributes sent in this Access-Accept will depends on radius-query service configuration.

   **Note** You use `aregcmd` to configure the attributes for the Access-Accept packet in the `AttributesToBeReturned` subdirectory under a radius-query service type.

4. If the Prime Access Registrar server finds a cache entry for the subscriber and if the EnableNotifications property is set to TRUE, the Prime Access Registrar server stores the client IP address in the subscriber’s cache.

5. If the Prime Access Registrar server receives an Accounting-Interim-Update packet from the GGSN, it responds by sending an Accounting-Response packet then sends the Accounting-Interim-Update packets to all the queried clients of the WAP Gateways.
If the WAP gateway queried clients do not respond to the Accounting-Interim-Update packets, the Prime Access Registrar server times out and retries by notifying the WAP gateways again. If there is no response after all the retries, the proxy packet is deleted and no change is made to the session or the WAP gateway’s state in the Prime Access Registrar server. The StaleSessionTimeout property under /Radius/Advanced is not applicable for Accounting-Interim-Update packets.

6. When the subscriber logs out of the network, the Prime Access Registrar server receives an Accounting-Stop packet and responds by sending an Accounting-Response back to the client. Before releasing the subscriber’s session, the Prime Access Registrar server looks for any client IP addresses in the subscriber’s cache. If it finds any, the Prime Access Registrar server sends Accounting-Stop packets to those clients with the attributes configured in the NotificationAttributeGroup subdirectory for each client.

The Prime Access Registrar server forms the attributes with those attributes in the session cache and from the Accounting-Stop packet. The Prime Access Registrar server uses the value configured for the Port property in the NotificationProperties subdirectory as the destination port for the Accounting-Stop packet and uses the client’s shared secret.

The Prime Access Registrar server then waits for Accounting-Response packets from each client to which it has sent Accounting-Stop packets. The Prime Access Registrar server waits for the time interval configured in the InitialTimeout property configured in the NotificationProperties subdirectory before sending another Accounting-Stop packet. If it does not receive an Accounting-Response packet, the Prime Access Registrar server sends additional Accounting-Stop packets until the number of attempts reaches the value configured in the MaxTries property in the NotificationProperties subdirectory.

7. When the Prime Access Registrar server receives an Accounting-Response packet from each client, the server releases the subscriber session.

If the Prime Access Registrar server does not receive Accounting-Response packets from all clients after the configured time and attempts, the server maintains the subscriber session for the time interval configured in the StaleSessionTimeout property in /Radius/Advanced then releases the subscriber session.

The Prime Access Registrar server maintains the subscriber session to address the quarantine IP address requirement. The Prime Access Registrar server must quarantine IP addresses if a WAP gateway does not respond to Accounting-Stop sent by the Prime Access Registrar server. The length of time an IP address is quarantined depends on the value of the InitialTimeOut property under the NotificationProperties subdirectory of /Radius/Clients/wap_gateway.

8. If the StaleSessionTimeout property is TRUE for a subscriber session, the Prime Access Registrar server rejects any query requests from clients for this session cache. After the StaleSessionTimeout expires, the Prime Access Registrar server will again send Accounting-Stop to all the clients listed in the session and proceeds to delete this subscriber session regardless of the status of the Accounting-Stop.

Configuration Examples

Note: In addition to the following configuration, the StaleSessionTimeout property must be set in /Radius/Advanced. This property has a default value of 1 hour.
The following shows an example configuration for a Query-Notify client:

[ //localhost/Radius/Clients/wap-gateway1 ]
Name = wap-gateway1
Description =
IPAddress = 10.100.10.1
SharedSecret = secret
Type = NAS
Vendor =
IncomingScript~ =
OutgoingScript~ =
EnableDynamicAuthorization = FALSE
NetMask =
EnableNotifications = TRUE
NotificationProperties/
    Port = 1813
    InitialTimeout = 5000
    MaxTries = 3
    NotificationAttributeGroup = notifyGroup

The following shows an example configuration for a Query-Notify AttributeGroup:

[ //localhost/Radius/Advanced/AttributeGroups/notifyGroup ]
Name = notifyGroup
Description =
Attributes/
    1. User-Name
    2. Acct-Session-Id
    3. NAS-Identifier
    4. NAS-Port

Memory and Performance Impact

Using the Query-Notify feature will have the following effects:

- There will be a memory impact because the Prime Access Registrar server caches IP addresses of clients queried in the session record.
- There will be an impact on performance because the Prime Access Registrar server has to persist the cached IP address information before responding to radius-query requests.

Support for Windows Provisioning Service

Prime Access Registrar supports Microsoft’s Windows Provisioning Service (WPS). WPS provides hotspot users with seamless service to public WLAN hotspots by using Microsoft Windows-based clients. The Microsoft WPS solution requires Microsoft-based software in the data center for the RADIUS server and the provisioning server.

This section contains the following topics:

- Call Flow
- Example Configuration
- Unsupported Features
Call Flow

The following is the WPS process and Wireless Internet Service Provider (WISP) packet sequence for a new wireless client login at a Wi-Fi hotspot location:

1. The client discovers the WISP network at a Wi-Fi hotspot.
2. The client authenticates as guest (with null username and credentials) to the Prime Access Registrar server.
3. The client is provisioned and a new account is created.
4. The client is authenticated using the new account credentials and accesses the Internet.

The Prime Access Registrar server performs the following functions during WPS:

1. Detects the guest subscriber login from the null username and null credentials during PEAPv0 (MS-PEAP) authentication.
2. Grants a successful login and returns a sign-up URL of the provisioning server as a PEAP-Type-Length-Value (TLV) in the next Access-Challenge Packet.

   The following is an example value for the URL PEAP-TLV:
   http://www.example.com/provisioning/master.xml#sign up

   Where #sign up is the parameter for this action and is a required element of the value.

   The sign-up URL value is passed when the user authenticates as guest. The sign-up URL is a fragment within the Master URL. You can also configure other fragments to be returned in the Master URL. See Master URL Fragments, page 17-11 for more information about the different fragments.

3. Sends a VLAN-ID or IP filter (or both) in the final Access-Accept packet to restrict the guest user’s accessibility to only the Provisioning server.
4. Authenticates using the user configuration in the user database after the client is provisioned and a new account is created.

Example Configuration

The following shows an example configuration for the WPS feature:

```
[ //localhost/Radius/Services/peapv0 ]
Name = peapv0
Description =
Type = peap-v0
IncomingScript~ =
OutgoingScript~ =
MaximumMessageSize = 1024
PrivateKeyPassword = <password>
ServerCertificateFile = <path_to_ServerCertificateFile>
ServerRSAKeyFile = <path_to_ServerRSAKeyFile>
CACertificateFile = <path_to_CACertificateFile>
CACertificatePath =<path_to_CACertificatePath>
ClientVerificationMode = Optional
VerificationDepth = 4
EnableSessionCache = True
SessionTimeout = "5 Minutes"
AuthenticationTimeout = 120
TunnelService = eap-mschapv2
EnableWPS = True
MasterURL = http://www.example.com/provisioning/master.xml
```
WPSGuestUserProfile = WPS-Guest-User-Profile

When you set the EnableWPS property to TRUE, you must provide values for the properties MasterURL and WPSGuestUserProfile. See Environment Variables, page 17-11 for more information.

Environment Variables

The following two environment variables are used to support WPS:

- Send-PEAP-URI-TLV
- Master-URL-Fragment

Send-PEAP-URI-TLV

Send-PEAP-URI-TLV property is a Boolean value used by the authenticating user service to make the PEAP-V0 service include the URI PEAP-TLV in the protected success message. Under different circumstances Prime Access Registrar might send back different fragments within the MasterURL to the client, as described above.

The conditions under which this has to be sent is best known to the user authentication service (the service that is specified within the eap-mschapv2 service, which in turn is the tunnel service for PEAP-V0 service). So when it decides that it needs to send back the URL it can set this variable to TRUE. The default value for this is FALSE.

Master-URL-Fragment

The Prime Access Registrar authenticating user service uses Master-URL-Fragment to set the fragment within the Master URL that needs to be sent back. The Prime Access Registrar user authentication service sets the fragment to different values under different circumstances. While the Send-PEAP-URL-TLV indicates whether to send the URL or not, Master-URL-Fragment is used to intimate which fragment within the URL needs to be sent. If this variable is not set and if it is required to send the URL, ‘#signup’ will be sent by default.

Master URL Fragments

The following sections describe the different fragments the RADIUS server might send to the AP in the Master URL:

- Sign up
- Renewal
- Password change
- Force update

Sign up

This value is passed when the user authenticates as guest. The following is an example value for the URL PEAP-TLV:

http://www.example.com/provisioning/master.xml#sign up

where #sign up is the parameter for this action and a required element of the value.
Renewal

This value is passed when the user’s account is expired and needs renewal before network access can be granted. The following is an example value for the URL PEAP-TLV:

http://www.example.com/provisioning/master.xml#renewal

where #renewal is the parameter for this action and a required element of the value.

Password change

This value is passed when the user is required to change the account password. An example value for the URL PEAP-TLV is:

http://www.example.com/provisioning/master.xml#passwordchange

where #passwordchange is the parameter for this action and a required element of the value.

Force update

This value is passed when the WISP requires the Wireless Provisioning Services on the client to download an updated XML master file. This method of updating the XML master file on the client should be used only to correct errors; otherwise, the TTL expiry time in the XML master file is used to provide background updates. The following is an example value for the URL PEAP-TLV:

http://www.example.com/provisioning/master.xml#forceupdate

where #forceupdate is the parameter for this action and a required element of the value.

Unsupported Features

The following features are part of the Microsoft WPS functionality, but are not supported in the Prime Access Registrar:

- Account Expiration and Renewal
- Password Changing and Force Update

Account Expiration and Renewal

When the user creates an account and logs in with that account, the RADIUS server authenticates and authorizes the request and sends back an Access-Accept with a Session-Timeout attribute. The Access Point (AP) then forces the wireless client to reauthenticate for every timeout value. When there is one timeout duration left in the user account, the RADIUS server needs to send back a renewal URL (a URL fragment within the master URL) to the client for the user to renew the account.

Prime Access Registrar does not support this feature because the interface the Prime Access Registrar server has with the CiscoSecure Remote Agent does not have provisions to get the expiration information of user account. However, this release does provide an environment variable to copy the URL fragment and to control whether or not to send the URL using another environment variable. This can be used to send the renewal URL. There are some limitations, however.
Password Changing and Force Update

The Password Changing option is passed when the user is required to change the account password. Force Update option is passed when the WISP requires the Wireless Provisioning Services on the client to download an updated XML master file.

These functions are not possible in this release for the same reason mentioned above, the loose coupling between Prime Access Registrar and the CiscoSecure Remote Agent. Additionally, there is no known use case for this. As mentioned above, you can use the newly added environment variables to trigger these options.

Command Completion

Prime Access Registrar’s command completion feature provides online help by listing possible entries to the current command line when you press the Tab key after entering a partial command. The Prime Access Registrar server responds based on:

- The location of the cursor including the current directory
- Any data you have entered on the command line prior to pressing the Tab key

The command completion feature emulates the behavior of Cisco IOS and Kermit. When you press the Tab key after entering part of a command, the Prime Access Registrar server provides any identifiable object and property names. For example, after you first issue `aregcmd` and log into Prime Access Registrar, enter the following:

```
cd <Tab>
```

Administrators/  Radius/

Pressing the Tab key consecutively displays possible context-sensitive choices.

In the following example, after changing directory to `/Radius/services/local-file` an administrator wants to see the possible types of authentication services that can set.

```
cd /Radius/services/local-file
```

```
//localhost/Radius/Services/local-file }
Name = local-file
Description =
Type = file
IncomingScript- =
OutgoingScript- =
OutagePolicy- = RejectAll
OutageScript- =
FilenamePrefix = accounting
MaxFileSize = "10 Megabytes"
MaxFileAge = "1 Day"
RolloverSchedule =

set type <Tab>
```

eap-leap    file    local    radius-session
eap-md5     group   odbc     rex
eap-sim     ldap    radius   tacacs-udp
```
Service Grouping Feature

Values can also be tab-completed. For example, if you decide to set the local-file service’s type to file, you can do the following:

```
set type f
```

and the command line completes to:

```
set type file
```

Service Grouping Feature

The Service Grouping feature enables you to specify multiple services (called subservices) to be used with authentication, authorization, or accounting requests. The general purpose is to enable multiple Remote Servers to process requests.

Perhaps the most common use of this feature will be to send accounting requests to multiple Remote Servers thus creating multiple accounting logs. Another common use might be to authenticate from more than one Remote Server where, perhaps the first attempt is rejected, other Remote Servers can be attempted and an Access-Accept obtained.

Clearly, in the accounting request example, each request must be successfully processed by each subservice in order for the originator of the accounting request to receive a response. This is known as a logical AND of each of the subservice results. In the authenticate example, the first subservice which responds with an accept is returned to the client or if all subservices respond with reject, then a reject is returned to the client. This is known as a logical OR of each of the subservice results.

A Service is specified as a Group Service by setting its type to group, specifying the ResultRule (AND or OR) and specifying one or more subservices in the GroupServices subdirectory. The subservices are called in numbered order and as such are in an indexed list similar to Remote Server specification in a radius Service. Incoming and outgoing scripts for the Group Service can be optionally specified.

A subservice is any configured non-Group Service. When a Group Service is used, each subservice is called in exactly the same manner as when used alone (such as if specified as the DefaultAuthenticationService). Incoming and Outgoing scripts are executed if configured and Outage Policies are honored.

This section contains the following topics:

- Configuration Example - AccountingGroupService
- Configuration Example 2 - AuthenticationGroupService

Configuration Example - AccountingGroupService

To configure an accounting Group Service to deliver accounting requests to multiple Remote Servers:

**Step 1**
The first task is to set up the subservices which are to be part of the AccountingGroupService. Since subservices are merely configured Services which have been included in a service group, you need only define two new Services.

For this example, we will define two new radius Services called OurAccountingService and TheirAccountingService. A provider might want to maintain duplicate accounting logs in parallel with their bulk customer’s accounting logs.

**Step 2**
Change directory to /radius/services. At the command line, enter the following:
cd /radius/services

[ //localhost/Radius/Services ]
Entries 1 to 2 from 2 total entries
Current filter: <all>
local-file/
local-users/

Step 3 At the command line, enter the following:

add OurAccountingService
add TheirAccountingService

The configuration of these Services is very similar to standalone Radius accounting service. Step-by-step configuration instructions are not provided, but the complete configuration is shown below:

[ //localhost/Radius/Services/OurAccountingService ]
Name = OurAccountingService
Description =
Type = radius
IncomingScript = OurAccountingInScript
OutgoingScript = OurAccountingOutScript
OutagePolicy = RejectAll
OutageScript =
MultipleServersPolicy = Failover
RemoteServers/
  1. OurPrimaryServer
  2. OurSecondaryServer

[ //localhost/Radius/Services/TheirAccountingService ]
Name = TheirAccountingService
Description =
Type = radius
IncomingScript = TheirAccountingInScript
OutgoingScript = TheirAccountingOutScript
OutagePolicy = RejectAll
OutageScript =
MultipleServersPolicy = Failover
RemoteServers/
  1. TheirPrimaryServer
  2. TheirSecondaryServer

The next step is to create the new AccountingGroupService. The purpose of this Service is to process Accounting requests through both OurAccountingService and TheirAccountingService.

Step 4 At the command line, enter the following:

add AccountingGroupService

Added AccountingGroupService

cd AccountingGroupService

[ //localhost/Radius/Services/AccountingGroupService ]
Name = AccountingGroupService
Description =
Type =
IncomingScript =
OutgoingScript =
Service Grouping Feature

set type group

Set Type group

**Step 5** Set the ResultRule to **AND** to ensure that both services process the accounting request successfully.

```
set ResultRule AND
```

Set ResultRule AND

```
ls
[ //localhost/Radius/Services/AccountingGroupService ]
Name = AccountingGroupService
Description =
Type = group
IncomingScript =
OutgoingScript =
ResultRule = AND
GroupServices/
```

```
set IncomingScript AcctGroupSvcInScript
```

```
set OutgoingScript AcctGroupSvcOutScript
```

Add OurAccountingService and TheirAccountingService as subservices of the Group Service.

**Step 6** At the command line, enter the following:

```
cd GroupServices
[ //localhost/Radius/Services/AccountingGroupService/GroupServices ]
```

```
set 1 OurAccountingService
```

Set 1 OurAccountingService

```
Set 2 TheirAccountingService
```

Set 2 TheirAccountingService

```
ls
[ //localhost/Radius/Services/AccountingGroupService ]
Name = AccountingGroupService
Description =
Type = group
IncomingScript = AcctGroupSvcInScript
OutgoingScript = AcctGroupSvcOutScript
ResultRule = AND
GroupServices/
  1. OurAccountingService
  2. TheirAccountingService
```
This completes the setup of the AccountingGroupService. To use this Service simply set it as the DefaultAccountingService and/or configure a policy/rule set which will select this Service. Essentially, this can be used in the same manner as any other standalone service.

**Summary of Events**

The following describes the flow of what happens when a client sends an accounting request which is processed by the AccountingGroupService:

1. ActGroupSvcInScript is executed.
2. OurAccountingService is called.
3. OurAccountingService's Incoming Script, OurAccountingInScript is called.
4. The request is sent to the Remote Server OurPrimaryServer and/or OurSecondaryServer, if necessary.
5. If a response is not received, because we used the AND ResultRule, the request failed and no response is sent to the client and the request is dropped. If a response is received, then the process continues.
6. OurAccountingService's Outgoing Script, OurAccountingOutScript is called.
7. TheirAccountingService is called.
8. TheirAccountingService's Incoming Script, TheirAccountingInScript is called.
9. The request is sent to the Remote Server TheirPrimaryServer and/or TheirSecondaryServer, if necessary.
10. If a response is not received, because we used the AND ResultRule, the request failed and no response is sent to the client and the request is dropped. If a response is received, then the process continues.
11. TheirAccountingService's Outgoing Script, TheirAccountingOutScript is called.
12. AcctGroupSvcOutScript is executed.

**Configuration Example 2 - AuthenticationGroupService**

To configure a Group Service for the purposes of providing alternate Remote Servers for a single authentication:

- **Note**
  - If Service A rejects the request, try Service B.

- **Step 1**
  The first task is to set up the subservices which are to be part of the AuthenticationGroupService. Since subservices are merely configured Services which have been included in a service group, we will simply define two new Services. For simplicity, we will define two new radius Services called AuthenticationServiceA and AuthenticationServiceB.

- **Step 2**
  At the command line, enter the following:

```
    cd /radius/services

    [ //localhost/RADIUS/services ]
```
Entries 1 to 2 from 2 total entries
Current filter: <all>
local-file/
local-users/

add AuthenticationServiceA

add AuthenticationServiceB

Step 3  The configuration of these Services is very similar to standalone Radius authentication service. Step-by-step configuration instructions are not provided, but the complete configuration is shown below:

[ //localhost/Radius/Services/AuthenticationServiceA ]
Name = AuthentictionServiceA
Description =
Type = radius
IncomingScript = AuthAInScript
OutgoingScript = AuthAOutScript
OutagePolicy = RejectAll
OutageScript = AuthAOutageScript
MultipleServersPolicy = Failover
RemoteServers/
  1. PrimaryServerA
  2. SecondaryServerA

[ //localhost/Radius/Services/AuthenticationServiceB ]
Name = AuthentictionServiceB
Description =
Type = radius
IncomingScript = AuthBInScript
OutgoingScript = AuthBOutScript
OutagePolicy = RejectAll
OutageScript = AuthBOutageScript
MultipleServersPolicy = Failover
RemoteServers/
  1. PrimaryServerB
  2. SecondaryServerB

The next step is to create the new “AuthenticationGroupService”. The purpose of this Service is to process authentication requests through both AuthenticationServiceA and AuthenticationServiceB if AuthenticationServiceA rejects the request.

Step 4  At the command line, enter the following:

add AuthenticationGroupService

Added AuthenticationGroupService
cd AuthenticationGroupService

[ //localhost/Radius/Services/AuthenticationGroupService ]
Name = AuthenticationGroupService
Description =
Type =
IncomingScript =
OutgoingScript =

set type group

Set Type group

Next set the ResultRule to OR because we want to ensure that if the first subservice rejects the request, we then try the second subservice. If the second subservice rejects the request, then the response to the client is a reject.

**Step 5**  At the command line, enter the following:

set ResultRule OR

Set ResultRule OR

Set IncomingScript AuthGroupSvcInScript

Set OutgoingScript AuthGroupSvcOutScript

Set IncomingScript AuthGroupSvcInScript

Set OutgoingScript AuthGroupSvcOutScript

Is

[ //localhost/Radius/Services/AuthenticationGroupService ]
Name = AuthenticationGroupService
Description =
Type = group
IncomingScript = AuthGroupSvcInScript
OutgoingScript = AuthGroupSvcOutScript
ResultRule = OR
GroupServices/

Now we must add the services we created "AuthenticationServiceA" and "AuthenticationServiceB" as subservices of the Group Service.

**Step 6**  At the command line, enter the following:

cd GroupServices

[ //localhost/Radius/Services/AuthenticationGroupService/GroupServices ]
Service Grouping Feature

Chapter 17      Using Cisco Prime Access Registrar Server Features

set 1 AuthenticationServiceA

    Set 1 AuthenticationServiceA

Set 2 AuthenticationServiceB

    Set 2 AuthenticationServiceB

Is

    [ //localhost/Radius/Services/AuthenticationGroupService ]
    Name = AuthenticationGroupService
    Description =
    Type = group
    IncomingScript = AuthGroupSvcInScript
    OutgoingScript = AuthGroupSvcOutScript
    ResultRule = OR
    GroupServices/
        1. AuthenticationServiceA
        2. AuthenticationServiceB

This completes the setup of the AuthenticationGroupService. To use this Service simply set it as the
DefaultAuthenticationService and/or configure a policy/rule set which will select this Service.
Essentially, this can be used in the same manner as any other standalone Service.

Summary of Events

The following describes the flow of what happens when a client sends an Authentication request which
is processed by the AuthenticationGroupService:

1. AuthGroupSvcInScript is executed.
2. AuthenticationServiceA is called.
3. AuthenticationServiceA’s Incoming Script, AuthAInScript is called.
4. If the response is a reject or the request is dropped (due to an Outage Policy):
   a. AuthenticationServiceA’s Outgoing Script, AuthAOutScript is called.
   b. Processing continues with the next service.
5. If the response is an Accept:
   a. AuthenticationServiceA’s Outgoing Script, AuthAOutScript is called.
   b. Skip to step 9.
6. AuthenticationServiceB is called.
7. AuthenticationServiceB’s Incoming Script, AuthBInScript is called.
8. Since this is the last subservice in our Group Service:
   a. AuthenticationServiceB’s Outgoing Script, AuthBOutScript is called.
SHA-1 Support for LDAP-Based Authentication

The Prime Access Registrar server supports secure hash algorithm (SHA-1) for LDAP-based authentication. This feature enables the Prime Access Registrar server to authenticate users whose passwords are stored in LDAP servers and hashed using the SHA-1 encoding scheme.

SHA-1 support actually adds functionality for the following three features to Prime Access Registrar:

- Authentication of PAP access requests against an LDAP user entry that uses the SHA-algorithm to hash the password attribute
- Authentication of PAP access requests against an LDAP user entry that uses the SSHA algorithm to hash the password attribute
- Configuration of the Prime Access Registrar server to dynamically determine how password attributes retrieved from LDAP are encrypted and process them accordingly

This enhancement is 100% backwards compatible. All previously supported values for the PasswordEncryptionStyle property are still supported and still provide the same behavior. The only noticeable change is that dynamic is now the default value for the PasswordEncryptionStyle property.

This section contains the following topics:

- Remote LDAP Server Password Encryption
- Dynamic Password Encryption
- Logs

Remote LDAP Server Password Encryption

Apart from the two values, none and crypt, of the PasswordEncryptionStyle property on a Remote LDAP Server, SHA-1 supports adds three additional values for the PasswordEncryptionStyle property. Table 17-2 lists the valid values for this property and describes the corresponding behavior.

<table>
<thead>
<tr>
<th>PasswordEncryptionStyle</th>
<th>Cisco Prime Access Registrar Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>All passwords retrieved from this LDAP server are assumed to be returned to Prime Access Registrar as clear text. (There is no change in this functionality.)</td>
</tr>
<tr>
<td>crypt</td>
<td>All passwords retrieved from this LDAP server are assumed to be returned to Prime Access Registrar as passwords encrypted using the UNIX crypt algorithm. (There is no change in this functionality.) Passwords can be preceded by the {crypt} prefix, which is stripped before comparing passwords.</td>
</tr>
</tbody>
</table>

b. Regardless of whether the request is Accepted or Rejected, processing will continue at step 9.


10. Standard processing continues.
SHA-1 Support for LDAP-Based Authentication

Chapter 17      Using Cisco Prime Access Registrar Server Features

Table 17-2      Remote LDAP Server Password Encryption Style Values (continued)

<table>
<thead>
<tr>
<th>PasswordEncryptionStyle</th>
<th>Cisco Prime Access Registrar Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHA-1</td>
<td>All passwords retrieved from this LDAP server are assumed to be returned to Prime Access Registrar as a Base64-encoded version of the user's password after it has been hashed using the SHA-1 mechanism (as defined by Netscape). Passwords can be preceded by the {sha} prefix, which is stripped before comparing passwords.</td>
</tr>
<tr>
<td>SSHA-1</td>
<td>All passwords retrieved from this LDAP server are assumed to be encrypted/hashed using the SSHA mechanism (as defined by Netscape). Passwords can be preceded by the {ssha} prefix, which is stripped before comparing passwords. Note This is a Netscape/iPlanet-specific mechanism.</td>
</tr>
<tr>
<td>dynamic</td>
<td>The value instructs Prime Access Registrar to choose the encryption mechanism on a case-by-case basis after it determines the presence of a known prefix, which the LDAP server prepends to the value of the password attribute. For example, if the following was returned from an LDAP server as a password attribute: {SHA}qZk+NkcGgWq6P1VxeFDCbJzQ2J0=, the password would be processed using the SHA-1 mechanism. This value will be the new default for the PasswordEncryptionStyle property.</td>
</tr>
</tbody>
</table>

Dynamic Password Encryption

When using the dynamic setting for the PasswordEncryptionStyle property on a Remote LDAP Server, the Prime Access Registrar server looks for the prefixes listed in Table 17-3 to determine if encryption or a hash algorithm should be used during password comparison.

Note Password prefixes are not case-sensitive.

Table 17-3      Remote LDAP Server Password Prefix Values

<table>
<thead>
<tr>
<th>Password Prefix</th>
<th>Encryption/Hash Algorithm Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>None; when no known prefix is found, the password attribute is assumed to be in clear text.</td>
</tr>
<tr>
<td>{crypt}</td>
<td>UNIX crypt algorithm</td>
</tr>
<tr>
<td>{sha}</td>
<td>Secure Hash Algorithm, version 1 (SHA-1)</td>
</tr>
<tr>
<td>{ssha}</td>
<td>SSHA-1, as defined by Netscape.</td>
</tr>
</tbody>
</table>

The default value for the PasswordEncryptionStyle property on a Remote LDAP Server is dynamic.
Using the dynamic setting for the PasswordEncryptionStyle property will require a bit more processing for each password comparison. When using dynamic, the Prime Access Registrar server must examine each password for a known prefix. This should have no visible impact on performance.

Logs

Turn on trace to level 4 to indicate (via the trace log) which password comparison method is being used.

Dynamic Attributes

Prime Access Registrar supports dynamic values for the configuration object properties listed below. Dynamic attributes are similar to UNIX shell variables. With dynamic attributes, the value is evaluated at run time. All of the objects that support dynamic attributes will have validation turned off in aregcmd.

This section contains the following topics:

- Object Properties with Dynamic Support
- Dynamic Attribute Format
- Configuration
- Example
- Notes
- Validation

Object Properties with Dynamic Support

The following object properties support dynamic values:

Radius

- DefaultAuthenticationService
- DefaultAuthorizationService
- DefaultAccountingService
- DefaultSessionManager
- IncomingScript
- OutgoingScript

Note: Do not use the following environment variables:


User-Profile for the BaseProfile, User-Group for the Group, User-Authorization for the AuthorizationScript, Session-Manager for the DefaultSessionManager, or Session-Service for the DefaultSessionService.
Dynamic Attributes

/Radius/Clients
   client1/
      IncomingScript
      OutgoingScript

/Radius/Userlist/Default
   user1/
      Group
      BaseProfile
      AuthenticationScript
      AuthorizationScript

/Radius/UserGroup
   Group1/
      BaseProfile
      AuthenticationScript
      AuthorizationScript

/Radius/Vendor
   Vendor1/
      IncomingScript
      OutgoingScript

/Radius/Service
   Service1/
      IncomingScript
      OutgoingScript
      OutageScript
      OutagePolicy

/Radius/RemoteServers
   remoteserver1/
      IncomingScript
      OutgoingScript

   Remoteldapserver1/
      Searchpath
      Filter

Note: To differentiate the properties that support dynamic attributes, we place a tilde (~) after each property, as in IncomingScript~. However, when the Prime Access Registrar administrator is required to set values for those properties, continue to use the original property name, such as set IncomingScript ${e|realm}{Test}. The tilde is only for visual effect, and including the tilde will generate an error (“310 command Failed.”)
Dynamic Attribute Format

The format of the dynamic attribute is:

$\{eq|attribute-name\}\{default-name\}$

where e stands for environment dictionary, q stands for request dictionary, and p stands for response dictionary. You can use e, q, and p in any order. The attribute name is the name for the attribute from environment dictionary, request dictionary, or response dictionary.

For example,

```plaintext
/Radius
DefaultAuthenticationService = $\{eq\|realm\}(local-users)$
```

The default Authentication Service is determined at run time. Prime Access Registrar first checks to see if there is one value of `realm` in the environment dictionary. If there is, it becomes the value of DefaultAuthenticationService. If there is not, check the value of realm in the request dictionary. If there is one value, it becomes the value of DefaultAuthenticationService. Otherwise, local-users is the DefaultAuthenticationService. If we do not set local-users as the default value, the DefaultAuthenticationService is `null`. The same concept applies to all other attribute properties.

The validation for the dynamic values of the object property will only validate the default value. In the above example, Prime Access Registrar will do validation to check whether local-users is one of services defined in the service subdirectory.

Note

When setting specific property values, do not use the tilde (~) in the property name. Doing so generates a 310 Command Failed error.

Tunneling Support Feature


Table 17-4 lists the tunneling attributes supported in this Prime Access Registrar release.

<table>
<thead>
<tr>
<th>Attribute Number</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>Tunnel-Type</td>
</tr>
<tr>
<td>65</td>
<td>Tunnel-Medium-Type</td>
</tr>
<tr>
<td>66</td>
<td>Tunnel-Client-Endpoint</td>
</tr>
<tr>
<td>67</td>
<td>Tunnel-Server-Endpoint</td>
</tr>
<tr>
<td>69</td>
<td>Tunnel-Password</td>
</tr>
<tr>
<td>81</td>
<td>Tunnel-Private-Group-ID</td>
</tr>
<tr>
<td>82</td>
<td>Tunnel-Assignment-ID</td>
</tr>
<tr>
<td>83</td>
<td>Tunnel-Preference</td>
</tr>
<tr>
<td>90</td>
<td>Tunnel-Client-Auth-ID</td>
</tr>
<tr>
<td>91</td>
<td>Tunnel-Server-Auth-ID</td>
</tr>
</tbody>
</table>

The tunneling attribute has the following format:
This section contains the following topics:

- **Configuration**
- **Example**
- **Notes**
- **Validation**

### Configuration

1. Configure the tag attributes as untagged attributes under the `/Radius/Advanced/Attribute Dictionary` directory (for example, `Tunnel-Type`).
2. Attach the "_tag" tag to these attributes when configuring the attributes under all of the other directories as tagged attributes (for example, `Tunnel-Type_tag10` under the `/Radius/Profiles/test` directory). Without the tag number, the default value is (_tag = _tag0).

### Example

```
/Radius/Advanced/Attribute Dictionary
/Tunnel-Client-ID
    Name = Tunnel-Client-Endpoint
    Description =
    Attribute = 66
    Type = STRING
    Min = 0
    Max = 253

/Radius/Profiles/test
    Name = test
    Description =
    /Attributes
        Tunnel-Client-Endpoint_tag3 = "129.56.112.1"
```

### Notes

1. "_tag" is reserved for the tunneling attributes. No other attributes should include this suffix.
2. The tag number value can range from 0 through 31.

### Validation

The Prime Access Registrar server checks whether the tag attributes are defined under the `/Radius/Advanced/Attribute Dictionary` directory. The server also checks whether the tag number falls within the range (0-31).
xDSL VPI/VCI Support for Cisco 6400

To provide this support, a distinction must be made between device authentication packets and regular user authentication packets. This section contains the following topics:

- Using User-Name/User-Password for Each Cisco 6400 Device
- Format of the New User-Name Attribute

Using User-Name/User-Password for Each Cisco 6400 Device

This approach assumes that for every 6400 NAS, a device-name/device-password is created for each. Following are the required changes:

For each NAS in Prime Access Registrar:

```
Name = test6400-1
Description =
IPAddress = 209.165.200.224
SharedSecret = secret
Type = NAS
Vendor =
IncomingScript =
OutgoingScript =
Device-Name = theDevice
Device-Password = thePassword
```

When the 6400 sends out the device authentication packet, it might have different User-Name/User-Password attributes for each 6400 NAS. When Prime Access Registrar receives the packet, it tries to obtain the Device-Name/Device-Password attributes from the NAS entry in the Prime Access Registrar configuration database. When the User-Name/User-Password in the packet match the configured Device-Name/Device-Password attribute values, Prime Access Registrar assumes that it must get the device. The next step is to replace the User-Name attribute with the concatenated <module>/<slot>/<port> string. From this point, the packet is treated as a regular packet.

A user record with the name of the concatenated string must be created.

Format of the New User-Name Attribute

After the device is identified, the User-Name attribute is replaced with the new value. This new value is the concatenation of 6400 <module>/<slot>/<port> information from the NAS-Port attribute and the packet is treated as a regular user authentication from this point on.

This format only supports NAS Port Format D. See Cisco IOS documentation for more information about NAS port formats.

The format of the new User-Name attribute is the printf of “%s-%d-%d-%d-%d-%d” for the following values:

- NAS-IP—in dot format of the NAS-Ip-Address attribute. For example, 10.10.10.10.
- slot—apply mask 0xF0000000 on NAS-Port attribute and shift right 28 bits. For example, NAS-Port is 0x10000000, the slot value is 1.
module—apply mask 0x08000000 on NAS-Port attribute and shift right 27 bits. For example, NAS-Port is 0x08000000, the module value is 1.

port—apply mask 0x07000000 on NAS-Port attribute and shift right 24 bits. For example, NAS-Port is 0x06000000, the port value is 6.

VPI—apply mask 0x00FF0000 on NAS-Port attribute and shift right 16 bits. For example, NAS-Port is 0x00110000, the VPI value is 3.

VCI—apply mask 0x0000FFFF on NAS-Port attribute. For example, NAS-Port is 0x00001001, the VCI value is 9.

### Apply Profile in Cisco Prime Access Registrar Database to Directory Users

You can define the **User-Profile** and **User-Group** environment variables in the directory mapping and Prime Access Registrar will apply the profiles defined in the Prime Access Registrar database to each directory user having any of these two variables set.

This section contains the following topics:
- User-Profile
- User-Group
- Example User-Profile and User-Group Attributes in Directory User Record

### User-Profile

This attribute is of type string with the format:

```
<Value1>::<Value2> ...
```

The **User-Profile** attribute is intended to hold a list of profile names. `<Value1>` and `<Value2>` represent the names of the profiles. They are separated by the `::` character, therefore, the `::` can not be part of the profile name. The order of values in the string has significance, as the profiles are evaluated from left to right. In this example, profile `<Value2>` is applied after profile `<Value1>`. Assume the user record has a field called `UserProfile` that holds the name of the profile that applies to this user. This field is mapped to the environment attribute **User-Profile**. Following is how the mapping is done with `aregcmd`:

```
QuickExample/
    Name = QuickExample
    Description =
    Protocol = ldap
    IPAddress = 209.165.200.224
    Port = 389
    ReactivateTimerInterval = 300000
    Timeout = 15
    HostName = QuickExample.company.com
    BindName =
    BindPassword =
    UseSSL = FALSE
    SearchPath = "o=Ace Industry, c=US"
    Filter = (uid=%s)
    UserPasswordAttribute = password
    LimitOutstandingRequests = FALSE
```


Directory Multi-Value Attributes Support

MaxOutstandingRequests = 0
MaxReferrals = 0
ReferralAttribute =
ReferralFilter =
PasswordEncryptionStyle = None
LDAPToEnvironmentMappings/
   UserProfile = User-Profile
LDAPToRadiusMappings/

After Prime Access Registrar authenticates the user, it checks whether User-Profile exists in the environment dictionary. If it finds User-Profile, for each value in User-Profile, Prime Access Registrar looks up the profile object defined in the configuration database and adds all of the attributes in the profile object to the response dictionary. If any attribute is included in more than one profile, the newly applied profile overrides the attribute in the previous profile.

User-Group

You can use the User-Group environment variable to apply the user profile as well. In Prime Access Registrar, a user can belong to a user group, and that user group can have a pointer to a user profile. When Prime Access Registrar finds that a packet has User-Group set, it obtains the value of the User-Profile within the user group, and if the User-Profile exists, it applies the attributes defined in the user profile to that user.

Note that in Prime Access Registrar, every user can also directly have a pointer to a user profile. Prime Access Registrar applies profiles in the following order:
1. If the user profile defined in the user group exists, apply it.
2. If the user profile defined in the user record exists, apply it.

The profile in User-Group is more generic than in User-Profile. Therefore, Prime Access Registrar applies the profile from generic to more specific.

Example User-Profile and User-Group Attributes in Directory User Record

You can use an existing user attribute in the user record to store profile info. When this is a new attribute, we suggest you create a new auxiliary class AR_UserRecord for whichever user class is used. AR_User_Profile and AR_User_Group are two optional members in this class. They are of type string. The mapping is as follows:

LDAPToEnvironmentMappings/
   AR_User_Profile = User-Profile
   AR_User_Group = User-Group

Directory Multi-Value Attributes Support

If any attributes mapped from the LDAP directory to the Prime Access Registrar response dictionary are multivalued, the attributes are mapped to multiple RADIUS attributes in the packet.
MultiLink-PPP (ML-PPP)

Prime Access Registrar supports MultiLink-PPP (ML-PPP). ML-PPP is an IETF standard, specified by RFC 1717. It describes a Layer 2 software implementation that opens multiple, simultaneous channels between systems, providing additional bandwidth-on-demand, for additional cost. The ML-PPP standard describes how to split, recombine, and sequence datagrams across multiple B channels to create a single logical connection. The multiple channels are the ports being used by the Network Access Server (NAS).

During the AA process, Prime Access Registrar authenticates the user connection for each of its channels, even though they belong to the same logical connection. The Authentication process treats the multilink connection as if it is multiple, single link connections. For each connection, Prime Access Registrar creates a session dedicated for management purposes. The session stays active until you logout, which subsequently frees up all of the ports in the NAS assigned to each individual session, or until the traffic is lower than a certain threshold so that the secondary B channels are destroyed thereafter. Prime Access Registrar has the responsibility of maintaining the active session list and discards any session that is no longer valid in the system, by using the accounting stop packet issued from NAS. The multiple sessions that were established for a single logical connection must be destroyed upon the user logging out.

In addition, the accounting information that was gathered for the sessions must be aggregated for the corresponding logical connection by the accounting software. Prime Access Registrar is only responsible for logging the accounting start and accounting stop times for each session. As those sessions belong to the same bundle, IETF provides two standard RADIUS attributes to identify the related multilink sessions. The attributes are Acct-Multi-Session-Id (attribute 50) and Acct-Link-Count (attribute 51), where Acct-Multi-Session-Id is a unique Accounting identifier used to link multiple related sessions in a log file, and Acct-Link-Count provides the number of links known to have existed in a given multilink session at the time the Accounting record was generated. The Accounting software is responsible for calculating the amount of the secondary B channel’s connection time.

The secondary B channel can go up and down frequently, based upon traffic. The Ascend NAS supports the Target-Util attribute, which sets up the threshold for the secondary channel. When the traffic is above that threshold the secondary channel is up, and when the traffic is below that threshold, the secondary B channel is brought down by issuing an Accounting stop packet to Prime Access Registrar. On the other hand, if you bring down the primary channel (that is, log out), the secondary B channel is also destroyed by issuing another Accounting stop packet to Prime Access Registrar.

Table 17-5 lists ML-PPP related attributes.

<table>
<thead>
<tr>
<th>Number</th>
<th>Attribute</th>
<th>Cisco NAS (IOS 11.3 Release)</th>
<th>Ascend NAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>44</td>
<td>Acct-Session-Id</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>50</td>
<td>Acct-Multi-Session-Id</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>51</td>
<td>Acct-Link-Count</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>62</td>
<td>Port-Limit</td>
<td>Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>124</td>
<td>Target-Util</td>
<td>Not Supported</td>
<td>Supported</td>
</tr>
<tr>
<td>125</td>
<td>Maximum-Channels</td>
<td>Supported</td>
<td>Supported</td>
</tr>
</tbody>
</table>

Following are sample configurations for ML-PPP:

```plaintext
/Radius
  /Profile
    /Default-ISDN-Users
```
Name = Default-ISDN-Users
Description =
Attributes/
    Port-Limit = 2
    Target-Util = 70
    Session-Timeout = 70

/Radius
/UserGroups
/ISDN-Users
    Name = ISDN-Users
    Description = "Users who always use ISDN"
    BaseProfile = Default-ISDN-Users
    Authentication-Script =
    Authorization-Script =

The **Port-Limit** attribute controls the number of concurrent sessions a user can have. The **Target-Util** attribute controls the threshold level at which the second B channel should be brought up.

## Dynamic Updates Feature

The Dynamic Updates feature enables changes to server configurations made using `aregcmd` to take effect in the Prime Access Registrar server after issuing the `save` command, eliminating the need for a server `reload` after making changes.

Table 17-6 lists the RADIUS object and its child objects. For each object listed, the **Add** and **Modify or Delete** columns indicate whether a dynamic update occurs after adding, modifying, or deleting an object or attribute. Entries in the **Add** and **Modify or Delete** columns also apply to child objects and child attributes of the objects listed, unless the child object is explicitly listed below the object, such as `/Radius/Advanced/Ports` or `/Radius/Advanced/Interfaces`.

<table>
<thead>
<tr>
<th><strong>Object</strong></th>
<th><strong>Add</strong></th>
<th><strong>Modify or Delete</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Radius</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>UserLists</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>UserGroups</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Policies</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Clients</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Vendors</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Scripts</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Services</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SessionManagers</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ResourceManagers</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Profiles</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Rules</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Translations</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>TranslationGroups</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
The Dynamic Updates feature is subject to the following limitations:

- Changes to the Ports or Interfaces objects are not dynamically updated. An `aregcmd reload` command must be issued for these changes to be propagated to the Prime Access Registrar server.

- Changes (modifications and deletions) to existing Session Manager and Resource Manager objects are not dynamically updated. An `aregcmd reload` command must be issued for these changes to be propagated to the Prime Access Registrar server. However, additions of new Session Manager and Resource Manager objects are dynamically updated. Active sessions and allocated resources are preserved in this case.

- Changes to the Prime Access Registrar configuration might not be immediately propagated to the server. Dynamic updates are only carried out in a safe environment (that is, when packets are not being processed and when packet processing can be delayed until the changes can be made on the server safely). Dynamic updates will yield to packet processing when appropriate, thus not significantly impacting server performance.

- Changes to SNMP require the Prime Access Registrar server to be restarted (`/etc/init.d/arservagt restart`).

### NAS Monitor

The ability to monitor when a NAS is down (really only unreachable from Prime Access Registrar) is provided by `nasmonitor`. This program will repeatedly query a TCP port at the specified IP address until the device (NAS) is reachable. If the NAS is not reachable after a period of time, a warning e-mail is sent; if the NAS is still not reachable after another period of time, a message is sent to Prime Access Registrar to release all sessions associated with that NAS. The port to query, the query frequency, the first time interval, the back-off time interval, and the E-mail address to send to are all configurable (with defaults); the only required parameter is the NAS IP address. This program will work for any device that has a TCP port open; it can either be run by hand, when desired, or put in a cron job. See `nasmonitor -h` for details.

<table>
<thead>
<tr>
<th>Object</th>
<th>Add</th>
<th>Modify or Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>RemoteServers</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Replication</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Advanced</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SNMP</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Ports</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Interfaces</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

**Table 17-6 Dynamic Updates Effect on Radius Server Objects (continued)**

You must have `tclsh` installed in `/usr/local/bin` to use `nasmonitor`. `tclsh` is part of the standard Tcl installation that can be downloaded from [http://www.scriptics.com](http://www.scriptics.com).
Automatic Information Collection (arbug)

You can use the script arbug to collect information about your Prime Access Registrar server. The results are collected into a tarball that can be e-mailed or ftped to Cisco as requested.

arbug collects all the relevant information needed to report a problem to Prime Access Registrar support. The goal of the arbug script is to make sure all the necessary information is collected.

Note

The arbug script neither updates nor replaces any system or Prime Access Registrar-related configuration.

This section contains the following topics:

- Running arbug
- Files Generated

Running arbug

To run the arbug script, change directory to /cisco-ar/bin and enter the following:

```
./arbug
```

The following is a typical sequence.

Looking around...
Cluster:
User: admin
Password:
The report /tmp/arbug.10085/arbug.tar is ready to send; you may want to compress it first using gzip or compress.
hostname user_name bin>

Files Generated

The arbug script generates five files that are compressed into a tarball. Table 17-7 provides a summary of the information found in each of the files.

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>car.debug.tar.*</td>
<td>Machine-specific information including OS type, RAM details, disk space information, swap space information, patch information and open file details.</td>
</tr>
<tr>
<td>car.config.tar.*</td>
<td>Prime Access Registrar server configuration, server statistics, database dump by taking the administrator username and password as the input.</td>
</tr>
<tr>
<td>car.confini.tar.*</td>
<td>Information about ODBC.ini files and SNMP configuration</td>
</tr>
<tr>
<td>car.core.tar.*</td>
<td>Core files if any are present</td>
</tr>
<tr>
<td>car.logcerscr.tar.*</td>
<td>Information from scripts directory, certificate directory, license directory</td>
</tr>
</tbody>
</table>
Simultaneous Terminals for Remote Demonstration

Multiple people can view and interact in a single demonstration by using the share-access program, a standard GNU release with a special configuration for use with Prime Access Registrar. To run screen, a technical support specialist (CSE or DE) will telnet to your server and log in as cisco. While you run /opt/CSCOar/bin/share-access (assuming /opt/CSCOar is the Prime Access Registrar path) as root, the CSE or DE runs /opt/CSCOar/bin/share-access -r root. Now both people (or more) can see what the other types, as well as the results of the commands entered. The special Prime Access Registrar configuration only allows root and cisco to run screen. To end a share-access session, type Control-D.

Support for RADIUS Check Item Attributes

Prime Access Registrar supports RADIUS check item attributes configuration at the user and group levels. You can configure the Prime Access Registrar server to check for attributes that must be present or attributes that must not be present in the Access-Request packet for successful authentication.

When using check item attributes, the Prime Access Registrar server will reject Access-Requests if:

- Any of the configured check item attributes are not present in the Access-Request packet
- Any of the Access-Request packet’s check item attribute values do not match with those configured check item attribute values

For remote servers using either LDAP or ODBC, Prime Access Registrar allows for mapping of certain LDAP or ODBC fields to check item attributes. The mapped attributes can be used as check item attributes while processing the Access-Request packets.

When you configure check item attributes at both the user and group levels, the Prime Access Registrar server first checks the attributes of the user level before those of the group level. The Prime Access Registrar server must first authenticate the user’s password in the Access-Request before validating the check item attributes.

The Prime Access Registrar server logs details about any rejected Access-Requests as a result of check items processing.

Configuring Check Items

You use aregcmd to configure check item attributes.

Configuring User Check Items

To configure UserList check item attributes:

Step 1  Log into the Prime Access Registrar server, and use aregcmd to navigate to //localhost/Radius/UserLists/default/bob.

    [ //localhost/Radius/UserLists/Default/bob ]
    Name = bob
    Description =
    Password = <encrypted>
    AllowNullPassword = FALSE
    Enabled = TRUE
    Group~ = PPP-users
    BaseProfile~ =
    AuthenticationScript~ =
User-Specific Attributes

The Prime Access Registrar server supports user-specific attributes which enables the Prime Access Registrar server to return attributes on a per-user or per-group basis without having to use profiles.
The Prime Access Registrar server includes a property called `HiddenAttributes` to the User and UserGroup object. The `HiddenAttributes` property contains a concatenation of all user-level reply attributes. The `HiddenAttributes` property is not displayed, nor can the value be set or unset using the command-line interface.

The order of application of attributes is as follows:

1. UserGroup Base Profile
2. UserGroup Attributes
3. User Base Profile
4. User Attributes

The value of the `HiddenAttributes` property is used dynamically to construct and populate a virtual attributes directory in the User object. All values from the Attributes directory will go into the `HiddenAttributes` property. This occurs transparently when the administrator issues a save command.

**Packet of Disconnect**

Prime Access Registrar supports the Packet of Disconnect (POD) feature that enables the Prime Access Registrar server to send disconnect requests (PODs) to a NAS so that all the session information and the resources associated with the user sessions can be released. Prime Access Registrar can also determine when to trigger and send the POD.

For example, when a PDSN handoff occurs during a mobile session, the new PDSN sends out a new access-request packet to Prime Access Registrar for the same user. Prime Access Registrar should detect this handoff by the change in NAS-Identifier in the new request and trigger sending a POD to the old PDSN if it supports POD. Prime Access Registrar also provides an option for administrator to initiate sending POD requests through the command-line interface (CLI) for any user session.

Prime Access Registrar forwards POD requests from external servers to the destination NAS.

This section contains the following topics:

- Configuring Packet of Disconnect
- Proxying POD Requests from External Servers
- CLI Options for POD

**Configuring Packet of Disconnect**

This section describes how to configure the POD feature in the following:

- Configuring the Client Object
- Configuring a Resource Manager for POD

**Configuring the Client Object**

You should enable POD for each client object that might want to send disconnect requests to those clients. You enable POD in a client object using the EnableDynamicAuthorization property. This property is set to FALSE by default when you create a client object. The following example shows the default configuration for a new client object, NAS1.
Packet of Disconnect

If the Prime Access Registrar server might send a POD to this client, set the EnableDynamicAuthorization property to TRUE. When you set this property to TRUE, the Prime Access Registrar server creates a DynamicAuthorizationServer subdirectory under the client object. The following example shows a newly created DynamicAuthorizationServer subdirectory:

```
[ //localhost/Radius/Clients/NAS1/DynamicAuthorizationServer ]
Port = 3799
DynamicAuthSharedSecret =
InitialTimeout = 5000
MaxTries = 3
PODAttributeGroup =
COAAttributeGroup =
```

The default port is 3799. You can change the port, if desired.

The property DynamicAuthSharedSecret is initially set to the same as value as the client's SharedSecret property when you set EnableDynamicAuthorization to TRUE. You can chose to configure a different secret for POD in this subdirectory.

The InitialTimeout property represents the number of milliseconds used as a timeout for the first attempt to send a POD packet to a remote server. For each successive retry on the same packet, the previous timeout value used is doubled. You must specify a number greater than zero, and the default value is 5000 (or 5 seconds).

The MaxTries property represents the number of times to send a proxy request to a remote server before deciding the server is offline. You must specify a number greater than zero, and the default is 3.

The PODAttributeGroup property points to a group of attributes to be included in a disconnect-request packet sent to this client.

You can create and configure the PODAttributeGroup in the `/Radius/Advanced/AttributeGroups/` directory. The default group contains commonly used POD attributes NAS-Port and Acct-Session-Id.

The COAAttributeGroup property is used with the Change of Authorization (CoA) feature, also known as hot-lining.

Configuring a Resource Manager for POD

Prime Access Registrar provides a resource manager type called session-cache. When you set a resource manager to session-cache, the resource manager's configuration contains a subdirectory called AttributesToBeCached. The following is an example Resource Manager set to type session-cache:

```
[ //localhost/Radius/ResourceManagers/PODresourceMgr ]
Name = PODresourceMgr
Description =
Type = session-cache
OverwriteAttributes = FALSE
AttributesToBeCached =
QueryMappings =
```

The attributes you configure under the `AttributesToBeCached` directory are cached in the session record during session management. The cached attributes are then sent in the disconnect-request for this session.

The `OverwriteAttributes` property indicates whether to overwrite the existing attributes if there are any in the session record. Since this resource manager can be invoked during Access-Request as well as Accounting-Start processing, the `OverwriteAttributes` can be used to control if the attributes cached during Access-Request processing can be overwritten with the attributes available during Accounting-Start processing.

The following is an example of a typical session-cache resource manager:

```
[ //localhost/Radius/ResourceManagers/RM-New ]
  Name = RM-New
  Description =
  Type = session-cache
  OverwriteAttributes = TRUE
  AttributesToBeCached/
    1. Framed-IP-Address
    2. CDMA-Correlation-ID
  QueryMappings/
```

The attributes used in the example can be added as an indexed list using `add` or `set` commands (in any order).

### Proxying POD Requests from External Servers

Prime Access Registrar can also proxy the disconnect requests received from external servers. To make Prime Access Registrar listen for external POD requests, the `ListenForDynamicAuthorizationRequests` property under `/Radius/Advanced` should be set to TRUE. The default value for this is FALSE. The default POD listening port is 3799. However this can be changed by configuring a new port of type `pod` under `/Radius/Advanced/Ports` and setting the new port number accordingly.

For security reasons, the source of a POD request should be configured as a remote server in Prime Access Registrar and the remote server should be configured to accept PODs. Set the property `AcceptDynamicAuthorizationRequests` to TRUE to do this. The default for this is FALSE. POD requests from unauthorized sources are silently discarded.

### CLI Options for POD

Prime Access Registrar has options for the `query-sessions` and `release-sessions` CLI commands that enable querying or releasing sessions based on the session’s age. Another option enables querying or releasing sessions based on any valid RADIUS attribute available in the user’s session record. This section contains the following topics:

- `query-sessions`
- `release-sessions`

#### query-sessions

The syntax for using `query-sessions with-Age` option is the following:

```
query-sessions <path> with-Age <value>
```
Where <path> is the path to the server, session-manager or resource manager and <value> is the minimum age of the session specified in minutes or hours with options M, Minutes, H or Hours. This command returns all sessions that are older than the given age value.

The syntax for using `query-sessions` with-`Attribute` option is the following:

```
query-sessions <path> with-Attribute <name> <value>
```

Where <name> is the RADIUS attribute name and <value> is the value of the attribute to be matched. This command returns the sessions where a session record contains and matches the attribute value specified in <value> field.

**release-sessions**

The syntax for using `release-sessions` with-Age option is:

```
release-sessions <path> with-Age <value>
```

Where, <path> is the path to the server, session-manager or resource manager and <value> is the minimum age of the session specified in minutes or hours with options M for Minutes, H for Hours. This command returns all sessions that are older than the given age value.

The syntax for using `release-sessions` with-`Attribute` option is:

```
release-sessions <path> with-Attribute <name> <value>
```

Where, <name> is the RADIUS attribute name and <value> is the value of the attribute to be matched. This command returns the sessions where a session record contains and matches the attribute value specified in <value> field.

A new option is also available for `release-sessions` command to enable an administrator to trigger sending a POD for a user after the session is released.

```
release-sessions <path> with-<type> <value> [send-pod]
```

Where, <path> is the path to the server, Session Manager, or Resource Manager and <type> is one of the following: NAS, User, IP-Address ID, or Age. The `release-sessions` command with an optional [send-pod] at the end results in Prime Access Registrar sending a POD request. The PoD requests are directed to port number configured in `/radius/clients/<client name>/DynamicAuthorizationServer/port`. By default it is set to 3799. To configure udp xxx, set the port value as:

```
/radius/clients/<client name>/DynamicAuthorizationServer/port = xxx
```

### Configuring Change of Authorization Requests

Prime Access Registrar supports Change of Authorization (CoA) requests as defined in Internet RFC 3576 that provides a way to change authorization status of users already logged on to the network. The CoA feature, also known as hot-lining, provides a wireless operator the ability to efficiently address issues with users that might otherwise be unauthorized to access packet data services. When a problem occurs that causes a user to be unauthorized to use the packet data service, a wireless operator can use the CoA feature to resolve the problem and return the user’s packet data services.
When a user is hot-lined, their packet data service is redirected to a hot-line application that notifies the user of issues that might be blocking their access to normal packet data services. Hot-lining provides users with a way to address the issues blocking their access, such as billing issues, a prepaid account that has been depleted, or an expired credit card.

The CoA feature provides an option to the wireless operator administrator to send CoA packets to the client device when a user needs to be hot-lined. When to send a CoA request to a user depends on the wireless operator’s site-specific policies.

**Configuring the Client Object**

You should enable CoA for each client object that might want to send CoA requests to those clients. You enable CoA in a client object using the EnableDynamicAuthorization property. This property is set to FALSE by default when you create a client object. The following example shows the default configuration for a new client object, NAS1.

```
[ //localhost/Radius/Clients/NAS1 ]
  Name = nas1
  Description =
  IPAddress =
  SharedSecret =
  Type = NAS
  Vendor =
  IncomingScript- =
  OutgoingScript- =
  EnableDynamicAuthorization = FALSE
```

If the Prime Access Registrar server might send a CoA request to this client, set the EnableDynamicAuthorization property to TRUE. When you set this property to TRUE, the Prime Access Registrar server creates a DynamicAuthorizationServer subdirectory under the client object. The following example shows a newly created DynamicAuthorizationServer subdirectory:

```
[ //localhost/Radius/Clients/NAS1/COA ]
  Port = 3799
  DynamicAuthSharedSecret =
  InitialTimeout = 5000
  MaxTries = 3
  PODAttributeGroup =
  COAAttributeGroup =
```

The default port is 3799. You can change the port, if desired.

The property DynamicAuthSharedSecret is initially set to the same as value as the client’s SharedSecret property when you set EnableDynamicAuthorization to TRUE. You can chose to configure a different secret for CoA in this subdirectory.

The InitialTimeout property represents the number of milliseconds used as a timeout for the first attempt to send a CoA packet to a remote server. For each successive retry on the same packet, the previous timeout value used is doubled. You must specify a number greater than zero, and the default value is 5000 (or 5 seconds).

The MaxTries property represents the number of times to send a proxy request to a remote server before deciding the server is offline. You must specify a number greater than zero, and the default is 3.

The COAAttributeGroup property points to a group of attributes to be included in a CoA request packet sent to this client.
You can create and configure the COAAttributeGroup in the `/Radius/Advanced/AttributeGroups/` directory. The default group is not set to any value by default. When an attribute group is configured, the Prime Access Registrar server includes the attributes in this group in a CoA request. The values for these attributes are fetched from the user’s session record.

The CoA attribute group configuration can be used with a session-cache Resource Manager. For example, any new attributes that are to be sent in a CoA request can be configured for caching by the session-cache Resource Manager so they will be available in the session record when it is to be sent in the CoA request.

The CoA request might also contain AV pairs from the optional profile name in the `query-session` CLI command used to send the CoA request. In a 3GPP2 scenario, a profile containing the Filter-Id attribute set to a value “Hot-Line Active” can be included when a user is to be hot-lined. This can be used as a hot-line profile possibly containing other attributes as desired by the wireless operator. Another profile might be defined containing the Filter-Id attribute with the value “Hot-Line Normal.” This profile can be used with the `query-session` CLI command to bring the user back to normal.

The CoA request packet sent by the Prime Access Registrar server conforms to internet RFC 3756. In response to a CoA request initiated by the Prime Access Registrar server, the client should respond with a COA-ACK if it is able to hot-line the user based on credentials available in the CoA request. If the client is unable to hot-line the user for any reason, the client can include an error-cause attribute with the appropriate reason in a COA-NAK packet.

The Prime Access Registrar server logs all CoA responses. If the Prime Access Registrar server does not receive a response to a CoA request within the timeout period, it will retransmit for the configured number of retries, then logs an error if no response is received.

The Prime Access Registrar server forwards proxied CoA requests sent by external servers to the destination NAS. The CoA requests are proxied based on the NAS-IP-Address in the incoming request. The proxied CoA requests from external servers are forwarded to the destination NAS only if the source IP address is configured to accept dynamic authorization requests. The responses received from the NAS (either COA-ACK or COA-NAK) are forwarded back to the source where the Prime Access Registrar server received the original proxy request.

---

**Dynamic DNS**

Prime Access Registrar supports the Dynamic DNS protocol providing the ability to update DNS servers. The dynamic DNS updates contain the hostname/IP Address mapping for sessions managed by Prime Access Registrar.

You enable dynamic DNS updates by creating and configuring new Resource Managers and new Remote Servers, both of type `dynamic-dns`. The dynamic-dns Resource Managers specify which zones to use for the forward and reverse zones and which Remote Servers to use for those zones. The dynamic-dns Remote Servers specify how to access the DNS Servers.

This section contains the following topics:

- Configuring Dynamic DNS
- Testing Dynamic DNS with `radclient`
Configuring Dynamic DNS

Before you configure Prime Access Registrar you need to gather information about your DNS environment. For a given Resource Manager you must decide which forward zone you will be updating for sessions the resource manager will manage. Given that forward zone, you must determine the IP address of the primary DNS server for that zone. If the dynamic DNS updates will be protected with TSIG keys, you must find out the name and the base64 encoded value of the secret for the TSIG key. If the resource manager should also update the reverse zone (IP address to host mapping) for sessions, you will also need to determine the same information about the primary DNS server for the reverse zone (IP address and TSIG key).

If using TSIG keys, use aregcmd to create and configure the keys. You should set the key in the Remote Server or the Resource Manager, but not both. Set the key on the Remote Server if you want to use the same key for all of the zones accessed through that Remote Server. Otherwise, set the key on the Resource Manager. That key will be used only for the zone specified in the Resource Manager.

Configuring the Dynamic DNS

To configure the dynamic-dns remote server:

Step 1  Launch aregcmd.
Step 2  Create the dynamic-dns TSIG Keys:

    cd /Radius/Advanced/DDNS/TSIGKeys
    add foo.com

This example named the TSIG Key, foo.com, which is related to name of the example DNS server we use. You should choose a name for TSIG keys that reflects the DDNS client-server pair (for example, foo.bar if the client is foo and the server is bar), but you should use the name of the TSIG Key as defined in the DNS server.

Step 3  Configure the TSIG Key:

    cd foo.com
    set Secret <base64-encoded string>

The Secret should be set to the same base64-encoded string as defined in the DNS server. If there is a second TSIG Key for the primary server of the reverse zone, follow these steps to add it, too.

Step 4  Use aregcmd to create and configure one or more dynamic-dns Remote Servers.
Step 5  Create the dynamic-dns remote server for the forward zone:

    cd /Radius/RemoteServers
    add ddns

This example named the remote server ddns which is the related to the remote server type. You can use any valid name for your remote server.

Step 6  Configure the dynamic-dns remote server:

    cd ddns
    set Protocol dynamic-dns
Dynamic DNS

set IPAddress 10.10.10.1 (ip address of primary dns server for zone)

set ForwardZoneTSIGKey foo.com

set ReverseZoneTSIGKey foo.com

If the reverse zone will be updated and if the primary server for the reverse zone is different than the primary server for the forward zone, you will need to add another Remote Server. Follow the previous two steps to do so. Note that the IP Address and the TSIG Key will be different.

You can now use aregcmd to create and configure a resource manager of type dynamic-dns.

Step 7 Create the dynamic-dns resource manager:

```
cd /Radius/ResourceManagers
add ddns
```

This example named the service ddns which is the related to the resource manager type but you can use any valid name for your resource manager.

Step 8 Configure the dynamic-dns resource manager.

```
cd ddns
set Type dynamic-dns
set ForwardZone foo.com
set ForwardZoneServer DDNS
```

Finally, reference the new resource manager from a session manager. Assuming that the example configuration was installed, the following step will accomplish this. If you have a different session manager defined you can add it there if that is appropriate.

Step 9 Reference the resource manager from a session manager:

```
cd /Radius/SessionManagers/session-mgr-1/ResourceManagers
set 5 DDNS
```

Note The Property AllowAccountingStartToCreateSession must be set to TRUE for dynamic DNS to work.

Step 10 Save the changes you have made.

Testing Dynamic DNS with radclient

After the Resource Manager has been defined it must be referenced from the appropriate Session Manager. You can use radclient to confirm that dynamic DNS has been properly configured and is operational.
Dynamic Service Authorization Feature

To test Dynamic DNS using radclient:

**Step 1** Launch `aregcmd` and set the trace to level 4.

```
aregcmd
```

Login to the Prime Access Registrar server as an administrative user.

```
trace 4
```

**Step 2** Launch `radclient`.

```
cd /opt/CSCOar/bin
radclient
```

**Step 3** Create an Accounting-Start packet

```
acct_request Start username
```

Example:

```
set p [ acct_request Start bob ]
```

**Step 4** Add a Framed-IP-Address attribute to the Accounting-Start packet

**Step 5** Send the Accounting-Start packet

```
$p send
```

**Step 6** Check the `aregcmd` trace log and the dns server to verify that the host entry was updated in both the forward and reverse zones.

Dynamic Service Authorization Feature

Typically, Prime Access Registrar does not allow sending another Access-Request to the remote server after the user is connected to the LDAP servers for user authentication. The Dynamic Service Authorization feature allows you to access external databases such as LDAP and Oracle first to know which remote servers authenticated services need to be relayed. This feature enables Prime Access Registrar to determine whether to send access-accept back to the client or to send another access-request to the remote server such as LDAP and Oracle. Prime Access Registrar is able to perform this activity multiple times in a single access-request.

Configuring Dynamic Service Authorization Feature

Configuring the dynamic service authorization involves:

- Setting Up the Environment Variable
- Configuring the Script for the Dynamic Service Authorization
Setting Up the Environment Variable

Before configuring the dynamic service authorization feature, you must set the following three environment variables in Prime Access Registrar:

- **Re-Authentication-Service**
  When the Re-Authentication-Service is set, the server directs the request to the specified reauthentication service for processing.

- **Re-Authorization-Service**
  When the Re-Authorization-Service is set, the server directs the request to the specified reauthorization service for processing.

- **Re-Accounting-Service**
  When the Re-Accounting-Service is set, the server directs the request to the specified reaccounting service for processing.

You can set the environmental variable by using scripts. See Writing the Script, page 11-2 for more information.

---

**Note**

When using the same service for reauthentication and reauthorization, a loop can occur in these services. The loop count, by default is 10. You can change the loop count using the **Dynamic-Service-Loop-Limit** environment variable.

Following is a sample procedure for setting the environment variable:

```tcl
proc dynamicservice { request response environ } {
    $environ put Re-Authentication-Service "local-users"
    $environ put Re-Authorization-Service "local-users"
}
```

You can append this procedure by copying it into `tclscript.tcl` that is located in `/opt/CSCOar/scripts/radius/tcl` directory, or to the location that you chose when you installed Prime Access Registrar. You can also use this procedure as a separate script file and configure the script accordingly. See Adding the CheckEap.tcl Script, page 15-16 for more information on configuring the TCL script.

**Configuring the Script for the Dynamic Service Authorization**

To configure the script for the dynamic service authorization:

---

**Step 1**

Launch `aregcmd`.

```
aregcmd
```

**Step 2**

Change directory to `/Radius/Scripts`.

```
cd /Radius/Scripts
```

**Step 3**

Enter `dynamicservice`.

**Step 4**

Change the directory to `dynamicservice`.

```
cd dynamicservice
```

You get the following output:

```
[ //localhost/Radius/Scripts/dynamicservice ]
Name = dynamicservice
Description =
Language =
```
Step 5  Set the Language property to TCL.

**Set Language TCL**

Step 6  Set the filename property to `tclscript.tcl`.

**Set Filename tclscript.tcl**

Step 7  Set the EntryPoint property to `dynamicservice`.

**Set EntryPoint dynamicservice**

The following is an example of the script configuration:

```bash
cd /Radius
set IncomingScript dynamicservice
[ //localhost/Radius ]
IncomingScript~ = dynamicservice
DefaultAuthenticationService~ = local-users
DefaultAuthorizationService~ = local-users
```

Step 8  Enter **Save** to save the configuration.

The following shows a sample trace:

```
10/30/2013 12:32:02.258: P577: Packet received from 127.0.0.1
10/30/2013 12:32:02.259: P577: Packet successfully added
10/30/2013 12:32:02.259: P577: Trace of Access-Request packet
10/30/2013 12:32:02.259: P577:  identifier = 9
10/30/2013 12:32:02.259: P577:  length = 61
10/30/2013 12:32:02.259: P577:  User-Name = bob
10/30/2013 12:32:02.259: P577:  NAS-Port = 2
10/30/2013 12:32:02.260: P577:  NAS-Identifier = localhost
10/30/2013 12:32:02.261: P577:  Using NAS: localhost (127.0.0.1)
10/30/2013 12:32:02.262: P577:  Request is directly from a NAS: TRUE
10/30/2013 12:32:02.262: P577:  Authenticating and Authorizing with Service local-users
10/30/2013 12:32:02.262: P577:  Getting User bob’s UserRecord from UserList Default
10/30/2013 12:32:02.263: P577:  user list user bob’s password matches
10/30/2013 12:32:02.263: P577:  Processing UserGroup PPP-users’s check items
10/30/2013 12:32:02.263: P577:  User bob is part of UserGroup PPP-users
10/30/2013 12:32:02.263: P577:  Merging UserGroup PPP-users’s BaseProfiles into response dictionary
10/30/2013 12:32:02.264: P577:  Merging BaseProfile default-PPP-users into response dictionary
10/30/2013 12:32:02.264: P577:  Merging attributes into the Response Dictionary:
10/30/2013 12:32:02.264: P577:  Adding attribute Service-Type, value = Framed
10/30/2013 12:32:02.264: P577:  Adding attribute Framed-Protocol, value = PPP
10/30/2013 12:32:02.264: P577:  Adding attribute Framed-Routing, value = None
10/30/2013 12:32:02.264: P577:  Adding attribute Framed-MTU, value = 1500
10/30/2013 12:32:02.264: P577:  Adding attribute Framed-Compression, value = VJ TCP/IP header compression
10/30/2013 12:32:02.264: P577:  Adding attribute Ascend-Idle-Limit, value = 1800
10/30/2013 12:32:02.265: P577:  Merging UserGroup PPP-users’s Attributes into response Dictionary
10/30/2013 12:32:02.265: P577:  Merging attributes into the Response Dictionary:
```
Remote Session Management

Prime Access Registrar sessions can also be stored on a remote database. This improves the overall scalability of the number of sessions that Prime Access Registrar can simultaneously handle. The remote session manager internally uses two ODBC remote servers, Internal-ODBC-Read-Server and Internal-ODBC-Write-Server. Configurations pertaining to these internal remoteservers can be done under /Radius/Advanced/RemoteODBCSessionServer.

For more information on how to configure the Remote ODBC Session Server, refer to “RemoteODBCSessionServer” section on page 4-73.

Note

Ensure that the length of fields such as Username, Session/Resource Manager name Session-Key, Query-Key and so on are limited to the value specified in the Table 17-8 while it is configured. Although the field length of entire session record is 3KB it is limited to 2KB. This is practically sufficient to hold all the session parameters as well as the cached attributes (if any).

Table 17-8 Schema Details

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>NUMBER(10)</td>
</tr>
<tr>
<td>SESSION_KEY</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>NAME</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>PER_USER_RM</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>PER_GROUP_RM</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>IP_RM</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>IP</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>SESSION_MANAGER</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>AC</td>
<td>NUMBER(10)</td>
</tr>
<tr>
<td>NAS</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>CACHE_RM</td>
<td>VARCHAR2(20)</td>
</tr>
<tr>
<td>Q_VALUE</td>
<td>VARCHAR2(20)</td>
</tr>
</tbody>
</table>
### Wx Interface Support for SubscriberDB Lookup

Prime Access Registrar supports Diameter Wx interface to fetch the authentication vectors from HSS required for EAP-SIM/EAP-AKA authentication.

The EAP-SIM and EAP-AKA authentication service is extended to generate a Diameter message Multimedia-Authentication-Request (MAR), with the subscriber identity (IMSI), to the HSS when it requires the authentication vectors. The HSS sends a Diameter Multimedia-Authentication-Answer (MAA) back containing the number of triplets/quintuplets.

The PreRequestTranslationScript, PostRequestTranslationScript, PreResponseTranslationScript, and PostResponseTranslationScript are the available scripting points to modify the RADIUS and Diameter packets while sending and receiving the packets to or from the HSS. For more information, see Table 9-1 for EAP-AKA and Table 10-6 for EAP-SIM details.

**Figure 17-1   Wx Interface Support for SubscriberDB lookup**

For more information on Wx interface, see the 3GPP TS 29.124 and TS 29.229 specifications.

### Configuration Examples

The following shows an example configuration for EAP-AKA:

```plaintext
[ //localhost/Radius/Services/eap-aka-wx ]
  Name = eap-aka-wx
  Description =
  Type = eap-aka
  AlwaysRequestIdentity = False
  EnableIdentityPrivacy = False
  PseudonymSecret = <encrypted>
```
PseudonymRenewtime = "24 Hours"
PseudonymLifetime = Forever
Generate3GPPCompliantPseudonym = False
EnableReauthentication = False
MaximumReauthentications = 16
ReauthenticationTimeout = 3600
ReauthenticationRealm =
AuthenticationTimeout = 120
QuintetGenerationScript~ =
UseProtectedResults = False
SendReAuthIDInAccept = False
SubscriberDBLookup = Diameter
DestinationRealm = mpc.com
PreRequestTranslationScript~ =
PostRequestTranslationScript~ =
PreResponseTranslationScript~ =
PostResponseTranslationScript~ =

The following shows an example configuration for EAP-SIM:

[ //localhost/Radius/Services/eap-sim-wx ]
Name = eap-sim-wx
Description =
Type = eap-sim
NumberOfTriplets = 2
UseSimDemoTriplets = False
AlwaysRequestIdentity = False
EnableIdentityPrivacy = False
PseudonymSecret = <encrypted>
PseudonymRenewtime = "24 Hours"
PseudonymLifetime = Forever
Generate3GPPCompliantPseudonym = False
EnableReauthentication = False
MaximumReauthentications = 16
ReauthenticationTimeout = 3600
ReauthenticationRealm =
TripletCacheTimeout = 120
AuthenticationTimeout = 120
UseProtectedResults = False
SendReAuthIDInAccept = False
SubscriberDBLookup = Diameter
DestinationRealm = hss.com
PreRequestTranslationScript~ =
PostRequestTranslationScript~ =
PreResponseTranslationScript~ =
PostResponseTranslationScript~ =
Smart Grid Solution Management

Prime Access Registrar provides identity and access management for the smart grid solutions on IPv6 (and IPv4) networks. This is achieved using the Elliptic Curve Cryptographic (ECC) based certificate validation and SNMP support for TACACS+.

For EAP services, in addition to RSA certificates, Prime Access Registrar supports verification of ECC certificates. ECC uses elliptic curves to encrypt data when creating keys which enables you to create shorter and stronger keys for better efficiency. This is achieved using the Cisco SSL library APIs.

TACACS+ supports ASCII,PAP, and CHAP Authentication type, login and enable services, and LDAP, OCI, and ODBC services in addition to Local service.

The client certificate files and RSA or ECC key file are available in /cisco-ar/pki as client-cert.pem and client-key.pem respectively. Both the files must be in “.PEM” format, since the certificate validation is done based on the extension of the files.

ECC certificate validation is used in the following authentication methods:

- EAP-FAST
- EAP-Transport Level Security (TLS)
- EAP-TTLS
- Protected EAP

Lawful Interception (LI) Support in Prime Access Registrar

Lawful Interception (LI) is a requirement placed upon service providers to provide legally sanctioned official access to private communications. With the existing Public Switched Telephone Network (PSTN), LI is performed by applying a physical tap on the telephone line of the target in response to a warrant from a Law Enforcement Agency (LEA). However, Voice over IP (VoIP) technology has enabled the mobility of the end-user, so it is no longer possible to guarantee the interception of calls based on tapping a physical line.

When a Law Interception Server (LIS) of the LEA requests the LI server to start monitoring a particular target, LI server sends the corresponding request to the Prime Access Registrar server. XML schema definition files are shared between Prime Access Registrar and Mediation Partner device for request and response messages. A local web service, which runs on the Prime Access Registrar server listens to the messages from the LI server.

Prime Access Registrar provides support for Intercept Access Point (IAP) for receiving the intercept/monitoring request for the subscriber whose “Access Associated” Communications Identifying Information (AA CmII) is to be intercepted and delivered to the LIS.

Table 17-9 provides the list of supported RADIUS and Diameter intercept requests from the LIS.
Lawful Interception (LI) Support in Prime Access Registrar

Chapter 17      Using Cisco Prime Access Registrar Server Features

Initiating Monitoring Process

When the ProvisionTarget/DiaProvisionTarget request is received from the LIS, Prime Access Registrar adds the respective user in the monitoring list and starts monitoring the user events.

Table 17-10 lists the events of the target user that are reported to LIS:

<table>
<thead>
<tr>
<th>Intercept Request (RADIUS)</th>
<th>Intercept Request (Diameter)</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProvisionTargetRequest</td>
<td>DiaProvisionTargetRequest</td>
<td>To start monitoring the target user</td>
</tr>
<tr>
<td>DeProvisionTargetRequest</td>
<td>DiaDeProvisionTargetRequest</td>
<td>To stop monitoring the target user</td>
</tr>
<tr>
<td>LinkUpdateRequest</td>
<td>DiaLinkUpdateRequest</td>
<td>To query the target user in the monitored list</td>
</tr>
<tr>
<td>ListTargetRequest</td>
<td>DiaListTargetRequest</td>
<td>To list all the users that are currently being monitored</td>
</tr>
</tbody>
</table>
### Table 17-10: Targeted User Events

<table>
<thead>
<tr>
<th>Events</th>
<th>Attributes (RADIUS)</th>
<th>Attributes (Diameter)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Attempt (for RADIUS) /</td>
<td>• CaseIdentity (M)</td>
<td>• CaseIdentity (M)</td>
</tr>
<tr>
<td>Diameter Access Attempt (for Diameter)</td>
<td>• IAPSystemIdentity (M)</td>
<td>• IAPSystemIdentity (M)</td>
</tr>
<tr>
<td></td>
<td>• TimeStamp (M)</td>
<td>• TimeStamp (M)</td>
</tr>
<tr>
<td></td>
<td>• SubscriberIdentity (M)</td>
<td>• SubscriberIdentity (M)</td>
</tr>
<tr>
<td></td>
<td>• AccessMethod (C)</td>
<td>• OriginHost (C)</td>
</tr>
<tr>
<td></td>
<td>• NetworkAccessNodeIdentity (C)</td>
<td>• AuthRequestType (C)</td>
</tr>
<tr>
<td></td>
<td>• ProtocolSignal (O)</td>
<td>• SessionIdentity (C)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• AuthApplID (C)</td>
</tr>
<tr>
<td>Access-Accept (for RADIUS) /</td>
<td>• CaseIdentity (M)</td>
<td>• ProtocolSignal (O)</td>
</tr>
<tr>
<td>Diameter Access-Accept (for Diameter)</td>
<td>• IAPSystemIdentity (M)</td>
<td>• OriginRealm (C)</td>
</tr>
<tr>
<td></td>
<td>• TimeStamp (M)</td>
<td>• TargetNetwork (O)</td>
</tr>
<tr>
<td></td>
<td>• SubscriberIdentity (M)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AccessMethod (C)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• NetworkAccessNodeIdentity (C)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• IPAddress (C)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AccessSessionIdentity (M)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AccessSessionCharacteristics (C)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• LocationInformation (C)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ProtocolSignal (O)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Events</td>
<td>Attributes (RADIUS)</td>
<td>Attributes (Diameter)</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------</td>
<td>-----------------------</td>
</tr>
</tbody>
</table>
| Access-Failed (for RADIUS) / DiameterAccess-Failed (for Diameter) | • CaseIdentity (M)  
• IAPSystemIdentity (M)  
• TimeStamp (M)  
• SubscriberIdentity (M)  
• IPAddress (C)  
• ReasonForTermination (C)  
• ProtocolSignal (O) | • CaseIdentity (M)  
• IAPSystemIdentity (M)  
• TimeStamp (M)  
• SubscriberIdentity (M)  
• OriginHost (C)  
• AuthRequestType (C)  
• SessionIdentity (C)  
• AuthApplID (C)  
• ProtocolSignal (O)  
• OriginRealm (C)  
• TargetNetwork (O)  
• ResultCode (C)  
• ReasonForTermination (C) |
| Access-Session-Start (for RADIUS) / DiameterAccess-Session-Start (for Diameter) | • CaseIdentity (M)  
• IAPSystemIdentity (M)  
• TimeStamp (M)  
• SubscriberIdentity (M)  
• AccessSessionIdentity (M)  
• IPAddress (C)  
• ProtocolSignal (O) | • CaseIdentity (M)  
• IAPSystemIdentity (M)  
• TimeStamp (M)  
• SubscriberIdentity (M)  
• OriginHost (C)  
• AuthApplID (C)  
• SessionIdentity (M)  
• AuthRecNo (C)  
• ProtocolSignal (O)  
• OriginRealm (C)  
• TargetNetwork (O) |
Table 17-10  Targeted User Events (continued)

<table>
<thead>
<tr>
<th>Events</th>
<th>Attributes (RADIUS)</th>
<th>Attributes (Diameter)</th>
</tr>
</thead>
</table>
| Access-Session-End (for RADIUS) / DiameterAccess-Session-End (for Diameter) | • CaseIdentity (M)  
• IAPSystemIdentity (M)  
• TimeStamp (M)  
• SubscriberIdentity (M)  
• AccessSessionIdentity (M)  
• IPAddress (C)  
• ReasonforTermination (C)  
• ProtocolSignal (O) | • CaseIdentity (M)  
• IAPSystemIdentity (M)  
• TimeStamp (M)  
• SubscriberIdentity (M)  
• OriginHost (C)  
• AuthApplID (C)  
• SessionIdentity (M)  
• AuthRecNo (C)  
• ProtocolSignal (O)  
• OriginRealm (C)  
• TargetNetwork (O)  
• ReasonForTermination (C) |
| Access-Rejected (for RADIUS) / DiameterAccess-Rejected (for Diameter) | • CaseIdentity (M)  
• IAPSystemIdentity (M)  
• TimeStamp (M)  
• SubscriberIdentity (M)  
• IPAddress (C)  
• ReasonforTermination (C)  
• ProtocolSignal (O) | • CaseIdentity (M)  
• IAPSystemIdentity (M)  
• TimeStamp (M)  
• SubscriberIdentity (M)  
• OriginHost (C)  
• AuthRequestType (C)  
• SessionIdentity (C)  
• AuthApplID (C)  
• ProtocolSignal (O)  
• OriginRealm (C)  
• TargetNetwork (O)  
• ResultCode (C)  
• ReasonForTermination (C) |

Note
The attribute with (M) represents mandatory, (O) represents optional, (C) represents conditionally available.

If 3GPP-IMSI is present in the incoming request packet, the following 3GPP-related attributes are also reported to the LI server:
• 3GPP-IMSI
• Called-Station-Id
• Calling-Station-Id
• 3GPP-PDP-Type
• SGSN-Address
Lawful Interception (LI) Support in Prime Access Registrar

- GGSN-Address
- 3GPP-IMSI-MCC-MNC
- 3GPP-NSAPI
- 3GPP-SGSN-MCC-MNC
- 3GPP-IMEISV

Stopping Monitoring Process
On receiving the DeprovisionTarget request from LIS, the target user is removed from the monitoring list.

Querying Target User Events
On receiving the LinkUpdate request on target user from LIS, the target user details are checked in the monitoring list and message is sent to LIS as listed below:

- If the specified user is not currently being monitored, a reply with reason-code indicating that the user is currently not targeted is sent.
- If the specified user is currently being targeted and is not logged into the network, a reply with status stating that the user is “inactive” in the network is sent.
- If the specified user is currently being targeted and is logged into the network, a reply with the following attributes is sent:
  - Case Identity (M)
  - IAP System Identity (M)
  - Time Stamp (M)
  - Subscriber Identity (M)
  - Access Method (C)
  - Network Access Node Identity (C)
  - IP address (C)
  - Access Session Identity (M)
  - Access Session Characteristics (C)
  - Location information (C)
  - Protocol Signal (O)

Viewing Monitored Users
On receiving the ListTarget request from LIS, a list of users that are currently being monitored are sent to LIS. The reply will contain a surveillance-target-count attribute indicating the count of the number of users being targeted and multiple instances of surveillance-target-identifier attribute having the real identifiers.

Intercept Response
Each request from the LIS contains a transaction-id which is copied on to the reply from Prime Access Registrar. For each request type there is an appropriate response type with appropriate return data.

Table 17-9 provides the list of Intercept results for RADIUS and Diameter.
Table 17-11 Intercept Results for RADIUS and Diameter

<table>
<thead>
<tr>
<th>Intercept Response (RADIUS)</th>
<th>Intercept Response (Diameter)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProvisionTargetResult</td>
<td>DiaProvisionTargetResult</td>
<td>An acknowledgment for the request with the same transaction ID. For information on the request, see Initiating Monitoring Process, page 17-51.</td>
</tr>
<tr>
<td>DeprovisionTargetResult</td>
<td>DiaDeProvisionTargetResult</td>
<td>An acknowledgment for the request with the same transaction ID. For information of the request, see Stopping Monitoring Process, page 17-55.</td>
</tr>
<tr>
<td>LinkUpdateResult</td>
<td>DiaLinkUpdateResult</td>
<td>For LinkUpdate, see Querying Target User Events, page 17-55.</td>
</tr>
<tr>
<td>ListTargetResult</td>
<td>DiaListTargetResult</td>
<td>For ListTarget, see Viewing Monitored Users, page 17-55</td>
</tr>
</tbody>
</table>

Configuring Lawful Intercept

Two scripts which are LawfulIntercept and RexLiScript are to be configured to run LawfulIntercept service in Prime Access Registrar. LawfulIntercept script should be configured in the server's incoming scripting point which is used to check the provisioned status of the user in the incoming access request. If the user is provisioned in the data store, Virtual-Server-Outgoing-Script will be executed after the server's outgoingscripting point.

InitEntryPoint of LawfulIntercept script writes the targeted list of users to a file while the server is stopping and reads the targeted users back to data store while the server is starting.

RexLiScript is configured in Virtual-Server-Outgoing-Script that sends events of the provisioned users to the LI service client.

Configuring the Lawful Intercept

To configure Lawful Intercept:

Step 1 Create the RexLiScript script object that will be set in Virtual-Server-Outgoing-Script point.

```plaintext
[ //localhost/Radius/Scripts/virtual ]
  Name = virtual
  Description =
  Language = rex
  Filename = libLiScript.so
  EntryPoint = RexLiScript
  InitEntryPoint = InitRexLiScript
  InitEntryPointArgs =
```

Step 2 Create the LawfulIntercept script object.

```plaintext
[ //localhost/Radius/Scripts/LiScript ]
  Name = LiScript
  Description =
  Language = Rex
  Filename = libLiScript.so
  EntryPoint = LawfulIntercept
  InitEntryPoint = RexInitialize
  InitEntryPointArgs = virtual
```
Step 3  set LawfulIntercept script object to ServerIncoming scripting point;

```plaintext
[ //localhost/Radius ]
  IncomingScript~ = LiScript
```

**Note**  The file 'libLiScript.so' comes up with Prime Access Registrar kit. You have to copy it into /cisco-ar/scripts/radius/rex/ path.

Step 4  Save the configuration:

```plaintext
save
```

Step 5  Reload the configuration:

```plaintext
reload
```

---

**TACACS+ Support for AAA**

TACACS+ (Terminal Access Controller Access-Control System Plus) is a terminal access control protocol for routers, switches, network access servers and other networked computing devices. The main goal of TACACS+ is to provide separate authentication, authorization and accounting services.

In Prime Access Registrar, TACACS+ supports authentication, command authorization, and accounting. The authentication support is available for login services with PAP, CHAP, and ASCII authentication types. It also tracks and maintains the executed command details in the command accounting database. Configuration is supported through the CLI/GUI and statistics are provided through CLI, GUI, and SNMP. TACACS+ supports the following Prime Access Registrar services:

- Local-users and Local-file service
- OCI
- ODBC
- LDAP

The following shows an example configuration for TACACS+:

```plaintext
[ /Radius/Clients/mytac ]
  Name = mytac
  Description =
  Protocol = tacacs-and-radius
  IPAddress = 10.77.123.57
  SharedSecret = <encrypted>
  Type = NAS
  Vendor =
  IncomingScript~ =
  OutgoingScript~ =
  EnableDynamicAuthorization = FALSE
  NetMask =
  EnableNotifications = FALSE
  EnforceTrafficThrottling = TRUE
```
Prime Access Registrar provides command authorization support to authorize the cmd mode commands. Command authorization is based on device access rules and the decision to authorize is based on command sets and conditions or expressions defined for the access rules. They determine whether to authorize a set of commands for the user or not.

If you enable TACACS+ command authorization for a service, you must define the following:

- Command sets—You must configure the list of commands with the arguments and the action to perform: permit or deny.
- Device access rules—You must configure the conditions or expressions and the command sets that are applicable to the access rule if the conditions are met.
- Service—You must enable the device access and associate the device access rules for the service.

When a packet enters the service, it selects the first device access rule and evaluates the condition. If the condition is met, then the service applies the device access rule for the request. If the command that is processed matches a command listed in the command set, the service decides on whether to permit the command for the user or not based on the permissions set up. See the example below.

<table>
<thead>
<tr>
<th>Device Access Rule</th>
<th>Condition</th>
<th>Command Set</th>
<th>Command</th>
<th>Arguments</th>
<th>Action</th>
</tr>
</thead>
</table>
| NewAccessRule      | Expr1 OR Expr2  
Where:  
Expr1 = user-name=bob  
Expr2 = nas-identifier=~/PGW*/  
OR = Conditional operator | cmdset1   | show      | *         | permit  |
|                    |           | enable      | ~/serial*/ | deny      |

In the above example, if one of the conditions user-name = bob or nas-identifier =~/PGW*/ is met, then the service applies the device access rule. If the processed command with its arguments matches one of the commands listed above, then the service permits or denies the command according to the setup.

**Note**  
Prime Access Registrar supports POSIX Extended Regular Expression (ERE) for command arguments and condition expressions value property.

Figure 17-2 shows the transaction flow for TACACS+ command authorization.
The following is an example configuration of device access rules and command sets configured for a local-users service:

```
[ //localhost/Radius ]
Name = Radius
Description = 
Version = 7.0
IncomingScript~ = 
OutgoingScript~ =
DefaultAuthenticationService~ = local-users
DefaultAuthorizationService~ = local-users
DefaultAccountingService~ = local-file
DefaultSessionService~ = 
DefaultSessionManager~ = session-mgr-1
UserLists/
UserGroups/
Policies/
Clients/
Vendors/
Scripts/
Services/
SessionManagers/
ResourceManagers/
Profiles/
Rules/
Translations/
TranslationGroups/
RemoteServers/
CommandSets/
```
TACACS+ Support for AAA

---

DeviceAccessRules/
FastRules/
Advanced/
Replication/

--> cd /r/DeviceAccessRules/

[ //localhost/Radius/DeviceAccessRules ]
Entries 0 to 0 from 0 total entries
Current filter: <all>

--> add d2
Added d2

--> cd d2

[ //localhost/Radius/DeviceAccessRules/d2 ]
Name = d2
Description =
CommandSetNames =
Conditions =
DefaultDeviceAction = PermitAll
ConditionExpressions/

--> set Conditions "A1 and A2"
Set Conditions "A1 and A2"

--> SET CommandSetNames "cm1, CM2"
Set CommandSetNames "cm1, CM2"

--> CD ConditionExpressions/

[ //localhost/Radius/DeviceAccessRules/d2/ConditionExpressions ]
Entries 0 to 0 from 0 total entries
Current filter: <all>

--> add a1
Added a1

--> add a2
Added a2

--> cd a1

[ //localhost/Radius/DeviceAccessRules/d2/ConditionExpressions/a1 ]
Name = a1
Description =
Attribute =
Value =

--> Set Attribute user-name
Set Attribute user-name

--> Set Value user*
Set Value user*
 --> cd ..

 [ //localhost/Radius/DeviceAccessRules/d2/ConditionExpressions ]
 Entries 1 to 2 from 2 total entries
 Current filter: <all>

 a1/
a2/

 --> cd a2

 [ //localhost/Radius/DeviceAccessRules/d2/ConditionExpressions/a2 ]
 Name = a2
 Description =
 Attribute =
 Value =

 --> Set Attribute user-group
 Set Attribute user-group

 --> Set Value ABC
 Set Value ABC

 --> cd /r/CommandSets/

 [ //localhost/Radius/CommandSets ]
 Entries 0 to 0 from 0 total entries
 Current filter: <all>

 --> add cm1
 Added cm1

 --> cd cm1

 [ //localhost/Radius/CommandSets/cm1 ]
 Name = cm1
 Description =
 Commands/

 --> cd Commands/

 [ //localhost/Radius/CommandSets/cm1/Commands ]

 --> Set 1 "permit show **"
 Set 1 "permit show **"

 --> cd ..

 [ //localhost/Radius/CommandSets/cm1 ]
 Name = cm1
 Description =
 Commands/

 --> cd ..

 [ //localhost/Radius/CommandSets ]
 Entries 1 to 1 from 1 total entries
 Current filter: <all>
cm1/

--> add cm2

Added cm2

--> cd cm2

[ //localhost/Radius/CommandSets/cm2 ]
Name = cm2
Description = Commands/

--> cd commands/

[ //localhost/Radius/CommandSets/cm2/Commands ]

--> Set 1 "deny show all"

Set 1 "deny show all"

--> sav

Validating //localhost...
Saving //localhost...

For more information on configuring the command sets and device access rules in the GUI, see the CommandSets, page 3-52 and DeviceAccessRules, page 3-53 sections in Chapter 3, “Using the Graphical User Interface.”
Directing RADIUS Requests

You can use the policy engine to determine the AAA services for processing a request packet based on the User-Name suffix, User-Name prefix, Calling-Station-ID, Called-Station-ID and Nas-IP-Address. You configure the policy engine through policies and rules.

This chapter contains the following sections:
- Configuring Policies and Rules
- Routing Requests
- Standard Scripts Used with Rules

Configuring Policies and Rules

A policy is a group of rules. Each rule consists of a set of conditions and corresponding services. A rule succeeds if all the conditions specified in the rule are satisfied. If a rule succeeds, the services indicated by its service attributes are used to process the packet. However, Prime Access Registrar defers packet processing until the policy succeeds.

This section contains the following topics:
- Configuring Policies
- Configuring Rules
- Wildcard Support
- Script and Attribute Requirements
- Validation
- Known Anomalies

Configuring Policies

You configure policies under /Radius/Policies. To enable the Prime Access Registrar server to use policies, you must first configure policy named SelectPolicy.

```
[ //localhost/Radius/Policies/SelectPolicy ]
Name = SelectPolicy
Description =
Grouping = rule1|rule2
```
The Grouping property of a policy determines which rules are to be evaluated and in which order. Rules are evaluated from left to right. Use the pipe (|) or ampersand (&) character to group rules.

**Note**

Before you can provide rules in the Grouping property, the rules must first be added to the configuration under `/Radius/Rules`.

The following are the Grouping property rules:

- If rules are grouped with the pipe character (`rule1|rule2`), all rules in the group are evaluated in sequential order until one of the rules succeeds. If any one of the rules in the policy succeeds, the policy succeeds.
- If rules are grouped with the ampersand character (`rule1&rule2&rule3`), all the rules listed are evaluated until one of the rules fails. For the policy to succeed, all the rules in the policy must succeed.

### Configuring Rules

You configure rules under `/Radius/Rules`. When you add a rule, provide the script that should be executed for the rule and the attributes to use if the rule succeeds. The script you specify must be defined under `/Radius/Scripts`, as shown in the following:

```bash
[ //localhost/Radius/Rules/rule1 ]
Name = rule1
Description =
Type = radius
Script- =
Attributes/
   Authentication-service = local-users
   Authorization-service = local-users
   Realm = @cisco.com

[ //localhost/Radius/Scripts/ExecRealmRule ]
Name = ExecRealmRule
Description =
Language = Rex
Filename = librexscript.so
EntryPoint = ExecRealmRule
InitEntryPoint =
InitEntryPointArgs =
```

### Wildcard Support

Prime Access Registrar supports limited wildcard functionality in rules for Realm, DNIS, and CLID attributes, specifically the asterisk (*) and question mark (?) characters. The asterisk matches any number of characters, including the null character. The question mark matches any single character, not including the null character. Prime Access Registrar also supports both wildcard characters in one pattern, as in `CLID = 180098?87*`.

**Note**

The realms should start with either the @ or # character. For example, `Realm=@cisco.com`.

- For an exact matching of the realm, you should configure the rule with the exact realm. For example, for an exact match to `abc@cisco.com`, you should use `Realm=abc@cisco.com`.
• If you use Realm=cisco.com (without any valid character), values such as xyz@us.cisco.com, xyz@uk.cisco.com, abc@cisco.com, and so on can also match and return a success.

The following is an example using the asterisk wildcard character used in a Rule named *rule1*:

```
[ //localhost/Radius/Rules/rule1 ]
Name=rule1
Description =
ScriptName = ExecRealmRule
Attributes/
    Authentication-Service = Local-Users
    Authorization-Service = Local-Users
    Realm = ~/@*cisco.com/
```

Rule *rule1* succeeds when the domain of the username in an access request matches the @*cisco.com* pattern. Each of the following is a good match: @us.cisco.com, @eng.cisco.com, and @cisco.com. With a match, the *ExecRealmRule* script sets Authentication-Service and Authorization-Service to Local-Users in the environment dictionary.

The following is an example using the "?" wildcard character in a Rule named *rule2*:

```
[ //localhost/Radius/Rules/rule2 ]
Name = rule2
Description =
ScriptName = ExecDNISRule
Attributes/
    Authentication-Service = Translation-Service
    Authorization-Service = Translation-Service
    DNIS = 1800345987?
```

Rule *rule2* succeeds if the Called-Station-Id attribute (DNIS) in the packet matches 1800345987?. Each of the following is a good match: 18003459876 and 18003459870, while 1800345987 is not. With a match, the *ExecDNISRule* script sets Authentication-Service and Authorization-Service to Translation-Service in the environment dictionary.

**Script and Attribute Requirements**

The following script and attribute requirements exist:

• **/Radius/Policies/SelectPolicy** is the first policy Prime Access Registrar applies.

• The characters "|" and "&" are reserved as logical operands in a Grouping definition; they cannot be included in a **/Radius/Rules** name.

• A space is not permitted between the logical operands and the rules in a Grouping definition.

• The scripts included in the rules must be defined under the **/Radius/Scripts** directory.

• The attributes included in the rules must be defined under the **/Radius/Advanced/Attribute** Dictionary directory.

• The rules included in the policies must be defined under the **/Radius/Rules** directory.
Validation

When policies are configured, Prime Access Registrar performs the following validations:

- Ensures the scripts included in the rules are defined under the `/Radius/Scripts` directory.
- Ensures the attributes included in the rules are defined under the `/Radius/Advanced/Attribute Dictionary` directory.
- Ensures the rules included in the policies are defined under the `/Radius/Rule` directory.

Known Anomalies

The following anomalies currently exist:

- Grouping expressions are not checked for validity.
- The use of parentheses to denote precedence is not supported in a Grouping definition.
- A check is not performed to determine whether a policy that is included within another policy is defined under the `/Radius/Policies` directory.

Routing Requests

Using the policy engine, Prime Access Registrar enables you to route requests based on attributes in access request packets. The following sections describe how to route requests based on different attributes:

- Routing Requests Based on Realm
- Routing Requests Based on DNIS
- Routing Requests Based on CLID
- Routing Requests Based on NASIP
- Routing Requests Based on User-Name Prefix
- Attribute Translation
- Time of Day Access Restrictions

Routing Requests Based on Realm

The Prime Access Registrar policy engine can process request packets based on the realm in the User-Name attribute.

In the following example, request packets with the User-Name attribute containing @abc.com as the suffix should be processed locally and the request packets with User-Name attribute containing @xyz.com should be proxied to a remote AAA Server.

```
[ //localhost/Radius/Policies ]
SelectPolicy/
   Name = SelectPolicy
   Description =
   Grouping = abcrule|xyzrule
```
The following SelectPolicy refers to two rules abcrule and xyzrule:

1. When a request packet arrives, Prime Access Registrar executes SelectPolicy beginning with abcrule to determine if the User-Name attribute contains @abc.com as the realm. If so, the abcrule is successful as is SelectPolicy, therefore the packet is processed locally.

2. If the User-Name attribute does not contain @abc.com as the realm, Prime Access Registrar executes xyzrule to determine if the User-Name attribute contains @xyz.com. If so, xyzrule is successful as is SelectPolicy. Hence the request is proxied to the remote server specified in xyz-service.

In this example, the rules are grouped using the | (or) operator. So all the rules specified in the grouping parameter will be executed until one of them succeeds.

```
[ //localhost/Radius/Rules ]
abcrule/
  Name = abcrule
  Description =
  Script- = ExecRealmRule
  Attributes/
    Authentication-Service = local-users
    Authorization-Service = local-users
    Realm = @abc.com

xyzrule/
  Name = xyzrule
  Description =
  Script- = ExecRealmRule
  Attributes/
    Authentication-Service = xyz-service
    Authorization-Service = xyz-service
    Realm = @xyz.com
```

The ExecRealmRule script matches the realm with the suffix in the User-Name attribute and sets the appropriate service for processing the packet. This is a standard script available with Prime Access Registrar. Prime Access Registrar can also be configured to set a particular kind of service for multiple realms. For example, the following configuration can be used if packets with @pqr.com or @klm.com should be processed using the same service klm-service.

```
[ //localhost/Radius/Rules ]
rulex/
  Name = rulex
  Description =
  Script- = ExecRealmRule
  Attributes/
    Authentication-Service = klm-service
    Authorization-Service = klm-service
    Realm = "@pqr.com"  "@klm.com"
```

### Routing Requests Based on DNIS

The Prime Access Registrar policy engine can process request packets differently based on the DNIS (Called-Station-Id) attribute in the request packet.

In the following example, request packets with the Calling-Station-Id attribute that contain 1111111 should be processed by abc-service, while request packets with the Called-Station-Id attribute that contain 2222222 or 3333333 should be processed using xyz-service.

```
[ //localhost/Radius/Policies ]
  SelectPolicy/
```

---

**Cisco Prime Access Registrar 7.0.1 User Guide**

18-5
Routing Requests

Routing Requests Based on CLID

The Prime Access Registrar policy engine can process request packets differently based on the CLID value in arriving request packets.

In the following example, the request packets with a Calling-Station-Id (CLID) attribute value of 1111111 should be processed by abc-service and the request packets with the CLID attribute value of 2222222 or 3333333 should be processed using xyz-service.

```plaintext
[ //localhost/Radius/Policies ]
SelectPolicy/
    Name = SelectPolicy
    Description =
    Grouping = abcrule|xyzrule

    The following SelectPolicy refers to two rules, abcrule and xyzrule:

1. When a request packet arrives, Prime Access Registrar executes SelectPolicy beginning with abcrule to determine if the CLID attribute contains 1111111. If so, the abcrule is successful as is SelectPolicy, and the packet is processed using abc-service.

2. If the CLID attribute does not contain 1111111, Prime Access Registrar executes the xyzrule to determine if the CLID attribute contains 2222222 or 3333333. If so, xyzrule is successful and hence SelectPolicy becomes successful and the packet is processed using xyz-service.
```
The **ExecCLIDRule** script matches the CLID value configured in Prime Access Registrar with the value in the CLID attribute of the request packet and sets the appropriate service for processing the packet. **ExecCLIDRule** is a standard script available with Prime Access Registrar.

### Routing Requests Based on NASIP

The Prime Access Registrar policy engine can process request packets differently based on the client IP address value in arriving request packets.

In the following example, arriving request packets with the NAS-IP-Address attribute value 1.1.1.1 should be processed by abc-service and arriving request packets with the NAS-IP-Address attribute value 2.2.2.2 should be processed using xyz-service.

```
[ //localhost/Radius/Rules ]
abcrule/
    Name = abcrule
    Description =
    Script~ = ExecCLIDRule
    Attributes/
        Authentication-Service = abc-service
        Authorization-Service = abc-service
        CLID = 1111111

xyzrule/
    Name = xyzrule
    Description =
    Script~ = ExecCLIDRule
    Attributes/
        Authentication-Service = xyz-service
        Authorization-Service = xyz-service
        CLID = "2222222" "3333333"
```

The following **SelectPolicy** refers to two rules, **abcrule** and **xyzrule**:

1. When a request packet arrives, Prime Access Registrar executes **SelectPolicy** beginning with abcrule to determine if the NAS-IP-Address attribute contains an IP address from the subnet 1.1.1.0/24. If so, the abcrule is successful as is SelectPolicy, and the packet is processed using abc-service.

2. If the NAS-IP-Address attribute does not contain an IP address from the subnet 1.1.1.0/24, Prime Access Registrar executes xyzrule to determine if the NAS-IP-Address attribute contains 2.2.2.2. If so, xyzrule is successful as is SelectPolicy, and the packet is processed using xyz-service.

```
[ //localhost/Radius/Policies ]
SelectPolicy/
    Name = SelectPolicy
    Description =
    Grouping = abcrule|xyzrule
```

```
[ //localhost/Radius/Rules ]
abcrule/
    Name = abcrule
    Description =
    Script~ = ExecNASIPRule
    Attributes/
        Authentication-Service = abc-service
        Authorization-Service = abc-service
        Client-IP-Address = 1.1.1.0
        Subnet-mask = 255.255.255.0

xyzrule/
    Name = xyzrule
    Description =
    Script~ = ExecNASIPRule
    Attributes/
        Authentication-Service = xyz-service
        Authorization-Service = xyz-service
        Client-IP-Address = 2.2.2.2
        Subnet-mask = 255.255.255.0
```
Routing Requests

Name = xyzrule
Description =
Script~ = ExecNASIPRule
Attributes/
  Authentication-Service = xyz-service
  Authorization-Service = xyz-service
  Client-IP-Address = 2.2.2.2

The **ExecNASIPRule** script matches the Client IP address configured in Prime Access Registrar with the value in the NAS-IP-Address attribute of the request packet and sets the appropriate service for processing the packet. **ExecNASIPRule** is a standard script available with Prime Access Registrar.

Routing Requests Based on User-Name Prefix

You can use the Prime Access Registrar policy engine to select a service based on the prefix in the User-Name attribute.

In the following example, request packets with a User-Name attribute that contains @abc.com as the suffix and cisco as the prefix should be processed using the service abc-service. A request packet with User-Name attribute containing cisco/bob@abc.com will be processed using abc-service.

```
[ //localhost/Radius/Policies ]
SelectPolicy/
  Name = SelectPolicy
  Description =
  Grouping = suffixrule & prefixrule

The following SelectPolicy refers to two rules, *suffixrule* and *prefixrule*:
1. When a request packet arrives, Prime Access Registrar executes SelectPolicy beginning with suffixrule to determine if the realm in the User-Name attribute contains @abc.com. If so, the suffixrule is successful. Since there is an “&” operator between the rules, the prefixrule must also succeed for the SelectPolicy to be successful.
2. The prefixrule is now processed to determine if the prefix in the User-Name attribute contains cisco. If so, the prefixrule is successful which makes SelectPolicy successful, and the AA service is set to the service specified in the prefixrule.

[ //localhost/Radius/Rules ]
abcrule/
  Name = suffixrule
  Description =
  Script~ = ExecRealmRule
  Attributes/
    Realm = @abc.com

prefixrule/
  Name = prefixrule
  Description =
  Script~ = ExecPrefixRule
  Attributes/
    Authentication-Service = abc-service
    Authorization-Service = abc-service
    Delimiters = @#&/
    Prefix = cisco
    StripPrefix = No
```
ExecPrefixRule script matches the prefix configured in Prime Access Registrar with the prefix in the User-Name attribute of the request packet and sets the appropriate service for processing the packet. ExecPrefixRule is a standard script available with Prime Access Registrar. See ExecPrefixRule for more information.

Attribute Translation

The attribute translation feature supports the RADIUS proxy enabling you to customize attribute filters so that RADIUS attribute value (AV) pairs can be inserted, deleted, or substituted.

For example, when a request is proxied from AAA Server on ISP A to AAA Server on ISP B, some AV pairs might be substituted (such as IP address) because they might not be valid on the ISP B network. Additionally, ISP B might return some vendor-specific attributes (VSAs) that are not applicable to ISP A’s network. Therefore, ISP A will substitute ISP B’s VSA value pairs for ISP A’s VSAs.

Two configuration points under the /Radius directory support this feature,

- Translations
- TranslationGroups
- Parsing Translation Groups

Translations

Under the /Radius/Translations directory, any translation to insert, substitute, or translate attributes can be added. The following is a sample configuration under the /Radius/Translations directory:

```plaintext
[ //localhost/Radius/Translations/T1 ]
  Name = T1
  Description =
  DeleteAttrs = Session-Timeout,Called-station-id
  Attributes/
    Calling-Station-id = 1232909
```

DeleteAttrs is the set of attributes to be deleted from the packet. Each attribute is comma separated and no spaces are allowed between attributes.

Under the /Radius/Translations/T1/Attributes directory, the attributes that should be inserted and the attributes that should be substituted are specified. These AV pairs are either added to the packet if not present already or replaced with the configured value.

TranslationGroups

Under the /Radius/TranslationGroups directory, translations can be grouped and applied to certain sets of packets, which are referred to in a rule.

The following is a sample configuration under the /Radius/TranslationGroups directory:

```plaintext
[ //localhost/Radius/TranslationGroups/CiscoIncoming ]
  Name = CiscoIncoming
  Description =
  Translations/
    1. T1
```

The translation group is referenced through the Prime Access Registrar policy engine in the /Radius/Rules/<RuleName>/Attributes directory.

- Incoming-Translation-Groups are set to a translation group (for example CiscoIncoming).
Outgoing-Translation-Groups are to set to another translation group (for example CiscoOutgoing).

The following is an example of setting up a rule for a translation group.

```plaintext
[ //localhost/Radius/Rules/ciscotranslationrule ]
Name = ciscotranslationrule
Description =
Script~ = ParseTranslationGroupsByRealm
Attributes/
    Incoming-Translation-Groups = CiscoIncoming
    Outgoing-Translation-Groups = CiscoOutgoing
    Realm = @cisco.com
```

The ciscoTranslationRule rule must be referred to in the /Radius/Policies directory, so the Prime Access Registrar policy engine can invoke this rule. If the pattern of Realm, DNIS, or CLID matches the one defined in the rule, Prime Access Registrar sets the environment variable Incoming-Translation-Groups to CiscoIncoming. By looking up the definition of CiscoIncoming, Prime Access Registrar applies all of the translations to the incoming packet (before it is proxied to the other server).

When the proxied packet comes back to the RADIUS server, Prime Access Registrar sets the environment variable, Outgoing-Translation-Groups to CiscoOutgoing.

DNIS, CLID, and Realm are supported for filtering packets. Prime Access Registrar provides the following scripts to facilitate filtering based on DNIS, CLID and Realm.

### Parsing Translation Groups

Prime Access Registrar provides three scripts that enable you to parse translation groups based on the DNIS, CLID or Realm attribute in an incoming packet. These scripts are:

- ParseTranslationGroupsByDNIS
- ParseTranslationGroupsByCLID
- ParseTranslationGroupsByRealm

In the following example, request packets containing @abc.com as the realm should be proxied to the remote server defined under abc-service. Before redirecting the request packet to the remote server, the Calling-Station-Id of the packet should be changed to 111.

```plaintext
[ //localhost/Radius/Policies ]
SelectPolicy/
    Name = SelectPolicy
    Description =
    Grouping = realmrule & translaterule
```

The following SelectPolicy refers to two rules, realmrule and translaterule:

1. When a request packet arrives, Prime Access Registrar executes SelectPolicy beginning with “realmrule” to determine if the realm in the User-Name attribute contains 1.1.1.1. If so, the realmrule is successful and the AA service is set to abc-service.
2. Next Prime Access Registrar executes the translaterule to change the CLID of the packet to 111.

```plaintext
[ //localhost/Radius/Rules/ciscotranslationrule ]
Name = ciscotranslationrule
Description =
Script~ = ParseTranslationGroupsByRealm
Attributes/
    Incoming-Translation-Groups = CiscoIncoming
    Realm = @cisco.com
```
Time of Day Access Restrictions

You can use the Prime Access Registrar policy engine to implement access restriction on users based on the time of day. The following are **ExecTimeRule** script that implements this functionality:

- **ExecTimeRule** can be used to check the time at which the request packet arrives and determine if access should be granted based on the authorization parameters for the user.

- If the rule succeeds, **ExecTimeRule** sets the **Acceptedprofiles** Environment dictionary variable to a profile or a set of profiles, as in the following:

  ```java
  Acceptedprofiles=Regularaccess::Highprivilegeaccess
  ```

  __Note__
  
  If more than one profile is to be added to the **Acceptedprofiles** variable, use two colons to separate them (::).

If the user is authenticated, the **Baseprofile** of the user is compared with the **Acceptedprofiles**. All the profiles that are in the **Baseprofile** and in **Acceptedprofiles** will be used as profiles while sending the response for the user.

For example, consider the following user configuration of user1:

```java
[ //localhost/Radius/UserLists/new/user1 ]
Name = user1
Description =
Password = <encrypted>
AllowNullPassword = FALSE
Enabled = TRUE
Group~ = regularusers
BaseProfile~ = highprivilegeaccess::readonlyaccess::regularaccess
AuthenticationScript~ =
AuthorizationScript~ =
UserDefined1 =
Attributes/
CheckItems/
```

The **Baseprofile** of the user1 has highprivilegeaccess, readonlyaccess and regularaccess. If the **Acceptedprofiles** of the user has regularaccess and highprivilegeaccess, the profiles regularaccess and highprivilegeaccess will be used while sending the response packet.

This section contains the following topics:
Setting Time Ranges in ExecTimeRule

ExecTimeRule accepts time range in the following format.

Set timerange "** * * * *

The first star indicates minutes and can be a value from 0-59. The second star indicates hours and can be a value from 0-23. The third star indicates day of the month and can be a value from 1-31. The fourth star indicates month and can be a value from 1-12. The fifth star indicates day of the week and can be a value from 0-6 where 0 indicates Sunday, 1 indicates Monday, and so on.

For example, to schedule a particular action to occur every Sunday during the month of December, use a command line like this:

Set timerange "** * * 12 0"

ExecTimeRule Example Configuration

This section provides a configuration example where a user, user1, is only authorized for PPP service between 10 AM and 6 PM. If a login occurs at any other time, user1 will be authorized only for telnet service.

Policies

[ //localhost/Radius/Policies ]
Entries 1 to 1 from 1 total entries
Current filter: <all>
SelectPolicy/
   Name = SelectPolicy
   Description =
   Grouping = ppprule|telnetrule

Rules

[ //localhost/Radius/Rules ]
Entries 1 to 2 from 2 total entries
Current filter: <all>
ppprule/
   Name = ppprule
   Description =
   Script~ = ExecTimeRule
   Attributes/
      acceptedprofiles = default-ppp-users
timerange = "* 10-18 * * *"
telnetrule/
   Name = telnetrule
   Description =
   Script~ = ExecTimeRule
   Attributes/
      acceptedprofiles = default-telnet-users
Reducing Overhead Using Policies to Group Rules

When you configure a large number of rules, the processing of request packets can be slow. For example, if you have 50 rules and only the last rule succeeds, the Prime Access Registrar server will have to check the preceding 49 rules before executing the rule that succeeds. You can reduce this overhead by using policies to group rules.

The following sample configuration, Prime Access Registrar must choose the AA service to be used for two domains, abc.com and xyz.com, based on the DNIS. You can do this by configuring different policies for different domains.
Policies

In the following configuration, SelectPolicy selects the policy to process packets with realm abc.com or xyz.com. Based on the realm that arrives in the request packet, abcrealmrule and xyzrealmrule decide whether to use abc-policy or xyz-policy to process the packets. abc-policy and xyz-policy are configured with rules to check for DNIS numbers in the respective domains and set the AA services appropriately.

```
entries 1 to 3 from 3 total entries
Current filter: <all>
SelectPolicy/
   Name = selectpolicy
   Description =
   Grouping = abcrealmrule|xyzrealmrule
abc-policy/
   Name = abc-policy
   Description =
   Grouping = abcDNISrule1|abcDNISrule2
xyz-policy/
   Name = xyz-policy
   Description =
   Grouping = xyzDNISrule1|xyzDNISrule2
```

Rules

```
entries 1 to 6 from 6 total entries
Current filter: <all>
abcrealmrule/
   Name = abcrealmrule
   Description =
   Script- = ExecRealmRule
   Attributes/
      policy = abc-policy
      realm = @abc.com
xyzrealmrule/
   Name = xyzrealmrule
   Description =
   Script- = ExecRealmRule
   Attributes/
      policy = xyz-policy
      realm = @xyz.com
abcDNISrule1/
   Name = abcDNISrule1
   Description =
   Script- = ExecDNISRule
   Attributes/
      Authentication-Service = abc1-service
      Authorization-Service = abc1-service
      DNIS = 1111111
abcDNISrule2/
   Name = abcDNISrule2
   Description =
   Script- = ExecDNISRule
   Attributes/
      Authentication-Service = abc2-service
      Authorization-Service = abc2-service
      DNIS = 2222222
xyzDNISrule1/
   Name = xyzDNISrule1
   Description =
```
Standard Scripts Used with Rules

Prime Access Registrar software includes the following scripts that you can use with the rules:

- `ExecRealmRule`
- `ExecDNISRule`
- `ExecCLIDRule`
- `ExecNASIPRule`
- `ExecPrefixRule`
- `ExecSuffixRule`
- `ExecTimeRule`
- `ParseTranslationGroupsByRealm`
- `ParseTranslationGroupsByDNIS`
- `ParseTranslationGroupsByCLID`

**ExecRealmRule**

Use the `ExecRealmRule` script to determine the Authentication service and Authorization service to be used to process the request packet based on the suffix (Realm) in the User-Name attribute. You configure the Realm for which the packet should be checked and the service to use in the Attributes subdirectory of a rule. The `ExecRealmRule` script supports multivalued attributes with which you can configure to check for multiple Realms.

For example, the following statement checks the request packet for three realms. If one of these three realms is found in the request packet, the `ExecRealmRule` script sets the attributes to the values listed in the Attributes subdirectory of the rule that references the `ExecRealmRule` script.

```plaintext
set Realm "@cisco.com" "@foo.com" "#bar.com"
```

**Note**

Only the characters @ and # can be used as delimiters in `ExecRealmRule`. 

```
Script- = ExecRealmRule
Attributes/
    Authentication-Service = xyz1-service
    Authorization-Service = xyz2-service
    DNIS = 6666666
xyzDNISrule2/
    Name = xyzDNISrule2
    Description =
    Script- = ExecRealmRule
    Attributes/
        Authentication-Service = xyz2-service
        Authorization-Service = xyz2-service
        DNIS = 7777777
```
Prior to Cisco Prime Access Registrar (Prime Access Registrar), ExecRealmRule was interpreted as a regular expression pattern and was evaluated accordingly. ExecRealmRule now does a simple case insensitive comparison by default of the value specified for the realm attribute for the realm of a username and optionally performs regular expression matching.

You can now specify a pattern using the following notation:

```
~/pattern/
```

Where pattern is a string of alpha-numeric characters that might include wild card characters, as in “@*cisco.com” to match patterns (realms) that end in cisco.com.

**Note**
The question mark (?) should not be used without a character pattern preceding it. Specifying ? as the first character might have undesirable results. (For regexp terminology, the question mark should be preceded by an atom.)

The **ExecRealmRule** script checks the request packet for the Realm and applies the values set for the following attributes:

- Authentication-Service
- Authorization-Service
- Policy

### ExecDNISRule

Use the **ExecDNISRule** script to determine the Authentication service and Authorization service to be used to process the request packet based on the Called-Station-Id (DNIS) attribute. The DNIS for which the packet should be checked and the services can be configured through the Policy Engine. The **ExecDNISRule** script supports multivalued attributes, by which you can configure multiple DNIS for checking.

For example, the following statement checks for a Calling-Station-Id of 1111111, 2222222, or 3333333. If one of the DNIS values is true, the script applies the values set for the Authentication-Service, Authorization-Service, and Policy attributes.

```
set DNIS “1111111” “2222222” “3333333”
```

### ExecCLIDRule

Use the **ExecCLIDRule** script with the Policy Engine to determine the Authentication service and Authorization service to be used to process the request packet based on the Calling-Station-Id (CLID) attribute. The CLID for which the packet should be checked and the services can be configured through the Policy Engine. The **ExecCLIDRule** supports multivalued attributes by which you can configure multiple CLID for checking.

For example, the following statement checks for Calling-Station-ID and applies Authentication-Service, Authorization-Service, and Policy.

```
set CLID “1111111” “2222222” “3333333”
```

The **ExecCLIDRule** script checks the request packet for the Calling-Station-Id and applies the values set for the following attributes:
Chapter 18      Directing RADIUS Requests

Standard Scripts Used with Rules

ExecNASIPRule

The Policy Engine references the **ExecNASIPRule** script to determine the AAA Services, Policy and Session Manager based on the Client-IP-Address and Subnet-Mask set in the Policy Engine. The **ExecNASIPRule** script supports multi-value attributes by which multiple you can configure the Client-IP-Address and Subnet-Mask in `aregcmd` for checking.

For example, the following statements check for Client-IP-Address and Subnet-Mask and applies Authentication-Service, Authorization-Service, Accounting-Service, Policy, and Session-Manager.

```
set Client-IP-Address "1.1.1.1" "2.2.2.2" "3.3.3.3"
set Subnet-Mask "255.255.255.0" "255.255.0.0" "255.0.0.0"
```

The **ExecNASIPRule** script checks the request packet for the Client-IP-Address and Subnet-Mask and applies the values set for the following attributes:

- Authentication-Service
- Authorization-Service
- Accounting-Service
- Policy
- Session Manager

ExecPrefixRule

The Policy Engine references the **ExecPrefixRule** to determine the authentication and authorization services based on the prefix in the User-Name attribute of the request packet and assigns the appropriate service for processing the packet.

Table 18-1 lists the **ExecPrefixRule** script attributes.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delimiters</td>
<td>A list of valid delimiters; you can use any character as a delimeter, such as @#/-.</td>
</tr>
<tr>
<td>Prefix</td>
<td>List of valid prefixes.</td>
</tr>
<tr>
<td>StripPrefix</td>
<td>Option to strip or not to strip the prefix from the User-Name. If you configure this attribute to YES, the ExecPrefixRule strips the prefix from the User-Name. If you configure this attribute to NO, the ExecPrefixRule does not strip the prefix from the User-Name. By default, this attribute is set to YES.</td>
</tr>
</tbody>
</table>
For example, if cisco/bob@abc.com is the User-Name attribute, the ExecPrefixRule script sets the Authentication-Service to abc-service and User-Name to:

- bob@abc.com when the StripPrefix attribute is set to YES.
- cisco/bob@abc.com when the StripPrefix attribute is set to NO.

You can configure the Prefix attribute in Prime Access Registrar using the aregcmd as follows:

```plaintext
set Prefix “cisco”
```

The Prime Access Registrar server does a case-insensitive comparison of the value specified for the prefix attribute of a username.

You can configure the Prefix by specifying a pattern using the following notation:

```plaintext
~/pattern/
[ //localhost/Radius/Rules/prefix/Attributes ]
```

```plaintext
Delimiters = #@-/
Prefix = ~/cis*/
```

Where a pattern is a string of alpha-numeric characters that can include wild card characters, as in “cis*” to match patterns (realms) that start with “cis”.

**Note**

If you specify / as the delimiter while configuring ExecPrefix Rule, you must configure the prefix as

```plaintext
Prefix =~/pattern/.
```

**Note**

The question mark (?) should not be used without a character pattern preceding it. Specifying ? as the first character might have undesirable results. (For regexp terminology, the question mark should be preceded by an atom.)

### ExecSuffixRule

The Policy Engine references ExecSuffixRule to determine the AAA services, policy and session managers based on the suffix (or realm) set in the Policy Engine. You can use aregcmd to configure ExecSuffixRule to support multivalued attributes, as in the following:

```plaintext
set Suffix “cisco.com” “abc.com” “domain.com”
```

In the User-Name `bob@abc.com`, ExecSuffixRule first checks for any of the configured delimiters in the User-Name. If there is a match, ExecSuffixRule checks for the configured suffix in the User-Name. If the suffix matches, ExecSuffixRule checks for the value of the StripSuffix variable. If StripSuffix is set to Yes, the suffix (including the delimiter) is stripped from the User-Name attribute of the Access Request.

Table 18-2 lists the ExecSuffixRule script attributes.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delimiters</td>
<td>A list of valid delimiters; you can use any character as a delimiter such as these: @#/</td>
</tr>
</tbody>
</table>

```
Chapter 18  Directing RADIUS Requests

Standard Scripts Used with Rules

The Prime Access Registrar server does a case-insensitive comparison of the value specified for the suffix attribute for the suffix of a username.

You can also specify a pattern using the following notation:

```
~//pattern/
```

Where pattern is a string of alpha-numeric characters that might include wild card characters, as in “@*cisco.com” to match patterns (realms) that end in cisco.com.

**Note**

The question mark (?) should not be used without a character pattern preceding it. Specifying ? as the first character might have undesirable results. (For regexp terminology, the question mark should be preceded by an atom.)

### Configuring Suffix and Prefix Policies

**Step 1**

Activate the Policy Engine by configuring SelectPolicy.

For example, the following script explains you how to set a suffix and prefix policy in the Grouping list.

```
--> cd selectPolicy/
```

```
[ //localhost/Radius/Policies/SelectPolicy ]
Name = SelectPolicy
Description =
Grouping = suffixrule&prefixrule
```

**Step 2**

Run the configuration rules for Prefix and Suffix.

For example, the suffix and prefix rule configuration do the following:

- points to the **ExecSuffixRule** script
- specifies the delimiters for which to scan
- specifies the suffixes for which to scan
- indicates whether to strip the suffix from the User-Name

```
[ //localhost/Radius/Rules ]
Enteries 1 to 2 from 2 total entries
Current filter: <all>
```

```
prefixrule/
Name = prefixrule
Description =
Type = radius
Script~ = ExecPrefixRule
Attributes/
    Authentication-Service = local-users
```
In this example, if `bob@abc.com` is the User-Name attribute, **ExecSuffiRule** strips the User-Name `bob@abc.com` and sets the User-Name environment variable to `bob` because StripSuffix is configured as `yes`.

**ExecTimeRule**

Use the **ExecTimeRule** script to implement access restriction on users based on time. The **ExecTimeRule** script checks the time at which the request packet arrives and based on that the authorization parameters for the user can be decided. Based on the time of the request packet if the rule succeeds then **ExecTimeRule** sets the environment variable, Acceptedprofiles to a profile or a set of profiles.

For example, the following statement checks for Timerange and applies AcceptedProfiles.

```
Acceptedprofiles=Regularaccess::Highprivilegeaccess
```

**ParseTranslationGroupsByRealm**

The Policy Engine references the ParseTranslationGroupsByReal script to determine the incoming and outgoing translation groups based on realm set in the Policy Engine. Use the ParseTranslationGroupsByReal script to add or filter attributes in request and response packets. The ParseTranslationGroupsByReal script supports multi-value attributes enabling you to configure to check for multiple Realms.

For instance, the following statement checks for three Realms. If True, the Policy Engine applies the values set for the Incoming-Translation-Group and Outgoing-Translation-Groups attributes.

```
set Realm "@cisco.com" "@foo.com" "@bar.com"
```

**ParseTranslationGroupsByDNIS**

This script is referenced from the Policy Engine to determine the incoming and outgoing translation groups based on DNIS set in the Policy Engine. This script can be used to add/filter attributes in request/response packets. This script supports multi-value attributes, by which multiple DNIS can be configured for checking.

For example, the following statement checks for Calling-Station-ID and applies Incoming-Translation-Groups and Outgoing-Translation-Groups.

```
set DNIS "1111111" "2222222" "3333333"
```
ParseTranslationGroupsByCLID

The Policy Engine references the ParseTranslationGroupsByCLID script to determine the incoming and outgoing translation groups based on CLID set in the Policy Engine. You can use the ParseTranslationGroupsByCLID script to add and filter attributes in request and response packets. The ParseTranslationGroupsByCLID script supports multi-value attributes, by which you can configure multiple CLIDs for checking.

For example, the following statement checks for the Calling-Station-ID and applies Incoming-Translation-Groups and Outgoing-Translation-Groups.

```
set CLID "1111111" "2222222" "3333333"
```

ParseTranslationGroupsByDNIS

The ParseTranslationGroupsByDNIS script is referenced from the policy engine to determine the incoming and outgoing translation groups based on DNIS set in the policy engine. The ParseTranslationGroupsByDNIS script can be used to add and/or filter attributes in request and response packets. The ParseTranslationGroupsByDNIS script supports multi-value attributes, by which multiple DNIS can be configured for checking.

For example, the following statement checks for the Calling-Station-ID and applies Incoming-Translation-Groups and Outgoing-Translation-Groups.

```
set DNIS "1111111" "2222222" "3333333"
```
Using FastRules to Process Packet Flow

While using rule policy engine and scripting points to process packet flow, you need to be familiar with programming languages, and create scripts to attach them to the Prime Access Registrar configuration. FastRules concept is an easier and efficient alternative to rule policy engine and scripting points.

FastRules provides a mechanism to easily choose the right authentication, authorization, accounting, and query service(s), drop, reject, or break flows, run a script, choose a session manager and/or a chain of fast rules required for processing a packet.

FastRules has the following capabilities:

- Provides maximum flexibility and ease in matching information in the incoming packets for choosing the appropriate service to apply
- Provides an option to match values in AVPs based on value ranges, exact match, and simple string comparisons using regex
- Provides easy and efficient alternative to rule/policy engine and scripting points for most common use cases—reduces the use of external scripts to choose an appropriate service

Figure 19-1 describes the workflow for FastRules.

FastRules also provides an option to use Internal Scripts as part of the workflow. Internal scripts allow you to add, modify, or delete attributes in the request, response, and environment dictionaries for RADIUS, Diameter, and TACACS+. For more information about internal scripts, see Internal Scripts, page 11-14.
Configuring FastRules

FastRules can be configured either through the GUI or through a CLI. For more information on configuring fast rules through the GUI, see FastRules page 3.54.

A sample CLI for configuring fast rules is given below:

```bash
[ //localhost/Radius/Fastrules]
RuleDefinitions\n  radroot\n  Name = radroot
  Desc =
  Protocol = Radius
  Condition = A1 AND (A2 OR A3)
  Success = Authen(mod1)--fr2-->Author(oci)
  Failure =
  Attributes\n  A1\n    Name = A1
    Attribute = Calling-station-id
    Value = 1.2.3.4
  A2\n    Name = A2
    Attribute = User-name
    Value = bob
  A3\n    Name = A3
    Attribute = User-name
    Value = BEGINS_WITH(jane)
fr1\n  Name = fr1
```

Figure 19-1 FastRules Workflow

![FastRules Workflow Diagram](image-url)
Configuring FastRules

Desc = Protocol = Radius
     Condition = Attr1
     Success = Authen(proxy)
     Failure = DROP
Attributes
     Attr1
     Name = Attr1
     Attribute = Calling-station-id
     Value = 2.3.4.5

fr2
     Name = fr2
     Desc = Protocol = Radius
     Condition = Attribute1
     Success = Authen(local)
     Failure = Attributes
     Attribute1
     Name = Attribute1
     Attribute = Status
     Value = Failure

Order
     Radius
     1. radroot
     2. fr1
     Diameter
     1.
     Tacacs
     1.

Services
     mod1
     Name = mod1
     Desc = Type = mod
     Service = ldap
Dictionary
     +env:Calling = req:Calling-Station-Id
     -req:Calling-Station-Id = ""
     +req:Calling-Station-Id = '5.6.7.8'
     +res:Calling-Station-Id = env:Calling
     #res:Called-Station-Id = "123456789"
Wireless Support

This chapter provides the following information about using Cisco Prime Access Registrar (Prime Access Registrar) for wireless support:

- Mobile Node-Home Agent Shared Key, page 20-1
- 3GPP2 Home Agent Support, page 20-3
- Session Correlation Based on User-Defined Attributes, page 20-5
- Managing Multiple Accounting Start/Stop Messages, page 20-6
- NULL Password Support, page 20-6
- 3GPP Compliance, page 20-7

Mobile Node-Home Agent Shared Key

In a mobile wireless environment, a Home Agent (HA) can request a Mobile Node-Home Agent (MN-HA) shared key from the home Prime Access Registrar RADIUS server during a mobile IP registration request (RRQ) from a Packet Data Serving Node (PDSN). Prime Access Registrar supports distribution of the shared key in this environment. Prime Access Registrar encrypts the shared key using MD5 encryption before sending the key back to the HA in an Access-Accept packet.

When an HA receives an RRQ from a PDSN, the HA authenticates the RRQ using a MN-HA shared key. If the HA does not have the MN-HA shared key, it retrieves the MN-HA shared key from the Prime Access Registrar server by sending an Access-Request packet containing the 3GPP2 VSA CDMA-MN-HA-SPI (SPI attribute). Prime Access Registrar then sends the CDMA-MN-HA-Shared-Key corresponding to the user if the user has been successfully authenticated.

This section contains the following topics:

- Use Case Example
- Configuring User Attributes

Use Case Example

When HA receives an RRQ from a PDSN, it authenticates the RRQ by using a MN-HA shared key. If the HA does not have the MN-HA shared key, it retrieves the MN-HA shared key from the Prime Access Registrar server by sending an Access-Request packet containing the 3GPP2 vendor-specific attribute (VSA) CDMA-MN-HA-SPI, the Security Parameter Index (SPI attribute).
The Prime Access Registrar server then sends the CDMA-MN-HA-Shared-Key corresponding to the user if the user has successfully authenticated subject to the following rules:

1. If there is an incoming SPI and no configured SPI, the Prime Access Registrar server authenticates the user as usual and does not include a configured shared-key (if there is one) in the reply.

2. If the incoming SPI does not match the configured SPI, the Prime Access Registrar server authenticates the user as usual, but does not include the configured shared-key (if there is one) in the reply.

3. If the incoming SPI matches the configured SPI, but there is no shared-key configured, the Prime Access Registrar server proceeds with normal authentication. Since there is no shared-key, it will not be included in the reply.

4. If the incoming SPI matches the configured SPI and a configured shared-key exists, the Prime Access Registrar server proceeds to encrypt the MCD5 shared-key and include it in the Access-Accept.

The key to including the shared key in an Access-Accept is in matching the values of the SPI attribute.

### Configuring User Attributes

Prime Access Registrar server supports user-specific attributes which enables the Prime Access Registrar server to return attributes on a per-user or per-group basis without having to use profiles.

**Configuring the User Attributes**

To configure a user with the CDMA-MN-HA-SPI VSA to request a MN-HA shared key:

**Step 1** Log into the Prime Access Registrar server and launch `aregcmd`.

Log in as a user with administrative rights such as user `admin`.

**Step 2** Change directory to the attribute directory of the user.

```
cd /Radius/UserLists/Default/bob/Attributes
```

**Step 3** Set the CDMA-MN-HA-SPI VSA to the appropriate shared-key value.

```
set CDMA-MN-HA-SPI 1124
```

**Step 4** Set the CDMA-MN-HA-SPI VSA to the appropriate shared-key value.

```
set CDMA-MN-HA-Shared-Key secret112
```

**Step 5** Validate and save your changes.

```
validate
save
```
3GPP2 Home Agent Support

The Prime Access Registrar server supports 3GPP2 home agents. This support enables mobile IP clients that authenticate through a Prime Access Registrar RADIUS server to be told which home agent they should use.

Every Mobile IP client has a home domain that is served by a group of Home Agents (HA). The Mobile IP client sets up a tunnel to one (and only one) HA during a session while it roams. Typically, the domain can be determined by the Mobile IP client's network access identifier (NAI).

Note

The NAI is the userID submitted by the client during PPP authentication. In roaming, the purpose of the NAI is to identify the user as well as to assist in the routing of the authentication request.

During the authentication and authorization phase for each Mobile IP client, the RADIUS server must decide which HA from a group of HAs should be chosen to serve the client. This is called dynamic HA assignment.

This section contains the following topics:

- Home-Agent Resource Manager
- Querying and Releasing Sessions
- Access Request Requirements
- New 3GPP2 VSAs in the Cisco Prime Access Registrar Dictionary

Home-Agent Resource Manager

Prime Access Registrar supports dynamic HA assignment with a new resource manager type called home-agent. You configure the home-agent resource manager with a list of IP addresses. The Prime Access Registrar server assigns those addresses to clients whose request dictionary has the right attributes to indicate that an assignment should be done. This is similar to the ip-dynamic resource manager.

Unlike the ip-dynamic resource manager, HAs are not exclusively allocated to an individual session but are shared among a set of sessions.

Load Balancing

The goal of dynamic HA assignment is to have load balancing among HAs. The Prime Access Registrar server achieves this by evenly distributing mobile clients among HAs. At the same time, the Prime Access Registrar server ensures that the same HA is always assigned to the same Mobile IP client for the same session.
Configuring the Home Agent Resource Manager
To create a new resource manager using the aregcmd command:

**Step 1** Use the cd command to change to the Radius /ResourceManagers level.

```
   cd /Radius/ResourceManagers
   [ //localhost/Radius/ResourceManagers ]
   Entries 0 to 0 from 0 total entries
   Current filter: <all>
```

**Step 2** Use the add command to specify the name of a resource manager to create.

```
   add home-agent-pool
   --> Added home-agent-pool
```

**Step 3** Use the cd command to change to the Radius /ResourceManagers/home-agent-pool level.

```
   cd home-agent-pool
   [ //localhost/Radius/ResourceManagers/home-agent-pool ]
   Name = home-agent-pool
   Description =
   Type =
```

**Step 4** Use the set command to set the resource manager type to home-agent.

```
   set type home-agent
```

**Step 5** Use the ls command to view the subdirectories under home-agent-pool.

```
   ls
   [ //localhost/Radius/ResourceManagers/home-agent-pool ]
   Name = home-agent-pool
   Description =
   Type = home-agent
   Home-Agent-IPAddresses/
```

**Step 6** Use the cd command to change to the Radius/ResourceManagers/home-agent-pool/Home-Agent-IPAddresses level.

```
   cd Home-Agent-IPAddresses
   [ //localhost/Radius/ResourceManagers/home-agent-pool/Home-Agent-IPAddresses ]
```

**Step 7** Use the add command to add a single IP address or a range of IP addresses.

```
   add 209.165.200.200-209.165.200.254
   --> Added 209.165.200.200-209.165.200.254
```

**Querying and Releasing Sessions**

The aregcmd program has been modified to support a new filter for query-session and release-session. You can use this filter to restrict a request (either query or release) to just the sessions with a given home-agent IP address. For example, consider the following command line.
Chapter 20      Wireless Support

Session Correlation Based on User-Defined Attributes

--> query-session /radius with-home-agent 10.10.10.1

This command line will return all sessions that have a home-agent resource equal to the IP address 10.10.10.1.

Querying sessions using aregcmd displays the home-agent resource in each session as:

HA ddd.ddd.ddd.ddd

where each ddd is a decimal number from 0-255.

Access Request Requirements

When the home-agent resource manager receives an Access-Request that contains a CDMA-HA-IP-Addr attribute, the home-agent resource manager checks the response dictionary to see if it already has a CDMA-HA-IP-Addr attribute. If it does, then the Mobile IP client has been assigned a HA address already and the resource manager does not need to do anything.

If the value of the CDMA-HA-IP-Addr attribute in the request dictionary is 0.0.0.0, the home-agent resource manager assigns a HA and puts a new CDMA-HA-IP-Addr attribute whose value is the IP address of the HA in the response dictionary.

If the value of the CDMA-HA-IP-Addr attribute is not 0.0.0.0, the Mobile IP client has been assigned a HA address already. The home-agent resource manager copies the attribute (with its value) from the request dictionary into the response dictionary.

The Prime Access Registrar server might select the session manager based on the domain (using the rule engine, dynamic properties, or scripting), and it allows each session manager to have its own home-agent resource manager.

New 3GPP2 VSAs in the Cisco Prime Access Registrar Dictionary

Prime Access Registrar supports 3GPP2 vendor-specific attributes (VSAs) in the vendor-specific dictionary in /Radius/Advanced/Attribute Dictionary.

Note

There is no planned support for the Accounting-Container (3GPP2/6) attribute because it has different syntax than other vendor-specific attributes (VSAs) and requires special processing.

Session Correlation Based on User-Defined Attributes

All the session objects are maintained in one dictionary keyed by a string.

You can define the keying material to the session dictionary through a newly introduced environment variable, Session-Key. If the Session-Key is presented at the time of session manager process, it will be used as the key to the session object for this session. The Session-Key is of type string. By default, the Session-Key is not set. It’s value should come from attributes in the incoming packet and is typically set by scripts. For example, CLID can be used to set the value of Session-Key.
Use the script UseCLIDAsSessionKey as defined in the script rexscript.c to specify that the Calling-Station-Id attribute that should be used as the session key to correlate requests for the same session. This is a typical case for 3G mobile user session correlation. You can provide your own script to define other attributes as the session key.

In the absence of the Session-Key variable, the key to the session will be created based on the string concatenated by the value of the NAS and the NAS-Port.

There is a new option with-key available in aregcmd for query-sessions and release-sessions to access sessions by Session-Key.

Managing Multiple Accounting Start/Stop Messages

Since the PDSN is aware when it sends a RADIUS stop followed by a start record, it inserts the new Session Continue attribute (3GPP2/48) into the stop record. The existence of the Session Continue attribute denotes that a start record will immediately be sent and the packet data session continues on the PDSN.

When Prime Access Registrar receives an accounting stop packet, the following two conditions trigger a release of a session and its resources:

1. There is no 3GPP2/48 Session Continue attribute in the stop packet and the number of accounting stops received is greater or equal to the starts received for this session
2. The 3GPP2/48 Session Continue attribute is present in the stop packet, but its value is zero (0)

Note One of the conditions above must be true to release the session and its resources.

NULL Password Support

Prime Access Registrar introduced a new Prime Access Registrar environment variable, Allow-NULL-Password. At authentication time, if the following three conditions are met, user authentication is bypassed:

1. Allow-NULL-Password environment variable is set to TRUE.
2. The User-Password or CHAP-Password must be NULL in the incoming request. (If it is not NULL, normal password checking will occur.)
3. A user record exists for this user.

By default, the Allow-NULL-Password environment variable is not set.

Note You should be aware of the security impact when using the NULL Password feature.
You can set this environment variable in three different ways:

1. For the user in local database, one new field `AllowNullPassword` is added in the user record. When Prime Access Registrar fetches a user record for authentication, if this field is set to TRUE and `Allow-NULL-Password` environment variable does not exist, it sets `Allow-NULL-Password` environment variable to TRUE.

2. If the user record is in LDAP database, then the `LDAPToEnvironmentMappings` must be defined to map an attribute in LDAP user record to `Allow-NULL-Password` environment variable.

3. Through scripting which allows the decision to be made based on runtime conditions, such as attributes in the access-request or policies.

### 3GPP Compliance

Prime Access Registrar supports 3GPP compliance by implementing the following (refer to RFC 29.273):

- SWa reference point between an untrusted non-3GPP IP access and a 3GPP AAA server/proxy. See SWa Access Authentication and Authorization, page 20-8.
- STa reference point between a trusted non-3GPP access and a 3GPP AAA server/proxy. See STa Access Authentication and Authorization, page 20-8.
- SWm reference point between an Evolved Packet Data Gateway (ePDG) and a 3GPP AAA server/proxy. See SWm Access Authentication and Authorization, page 20-9.
- SWx reference point between a Home Subscriber Server (HSS) and a 3GPP AAA server. See SWx Authentication Procedure, page 20-10.

This topic also contains the following sections:

- 3GPP Call Flows, page 20-11
- This topic contains the following sections:, page 20-11
Figure 20-1 depicts the various interfaces used for 3GPP compliance in a mobile network.

**Figure 20-1  3GPP Interfaces**

**SWa Access Authentication and Authorization**

The SWa reference point is defined between a non-3GPP IP access and a 3GPP AAA server or between a non-3GPP IP access and a 3GPP AAA proxy.

The SWa access authentication and authorization procedure includes the following steps:

1. The 3GPP AAA server issues an unsolicited re-auth request towards the untrusted non-3GPP access, indicating that both re-authentication and re-authorization of the user is needed.

2. Upon receipt of such a request, the untrusted non-3GPP access responds to the request and indicates the disposition of the request. This procedure is mapped to the Diameter command codes Re-Auth-Request and Re-Auth-Answer.

3. Upon receiving the re-auth request, the untrusted non-3GPP access immediately invokes the SWa authentication and authorization procedure requesting the identity of the user through EAP and using DER/DEA commands, with the same session-ID.

4. If the re-authentication of the user is not successful, the untrusted non-3GPP access detaches the user.

**STa Access Authentication and Authorization**

The STa reference point is defined between a non-3GPP access network and a 3GPP AAA Server or between a non-3GPP access network and a 3GPP AAA Proxy.

Prime Access Registrar decides whether a non-3GPP access network is trusted or untrusted by using the access authentication and authorization procedure executed between the non-3GPP access network and the 3GPP AAA server. This is implemented by the STa and SWa reference points sharing the same Diameter application and partly sharing the same authentication and authorization procedure. The STa
and SWa reference points are clearly distinguished after the exchange of the first authentication and authorization messages, during which trusted/untrusted decision is made by the 3GPP AAA server and this decision is communicated to the non-3GPP access network.

The trusted non-3GPP access authentication and authorization requires DiaEAP with EAP-AKA or EAP-AKA'. Prime Access Registrar implements the STA access authentication and authorization procedure based on the mobility parameters transported by the non-3GPP access network to the 3GPP AAA server.

This procedure follows the SWa authentication and authorization procedure, with the following differences:

- Information elements that reflect information about the user’s service request and about the access network are mandatorily included in the authentication and authorization request.
- The information elements that describe the user’s subscription profile are downloaded to the non-3GPP access network.

### SWm Access Authentication and Authorization

The SWm reference point is defined between the ePDG and the 3GPP AAA server or between the ePDG and the 3GPP AAA proxy. It is used to authenticate and authorize a UE by transporting mobility parameters that are needed for the S2b interface. In particular this information may include the Packet Data Network (PDN) GW identity(s) and Access Point Name (APN(s)) currently allocated to a UE during a previous attach in a 3GPP access.

The SWm reference point performs authentication and authorization based on the reuse of the DER/DEA command set defined in the Diameter EAP application. The SWm access authentication and authorization procedure includes the following steps:

- The UE transmits a ‘tunnel establishment request’ message to the ePDG in order to establish a connection to the PDN.
- The ePDG initiates access authentication and authorization request to the 3GPP AAA server.
- During the access authentication and authorization procedure, the ePDG provides mobility parameters of the UE to the 3GPP AAA Server.
- The 3GPP AAA server performs IP mobility mode selection appropriately and upon successful authorization, it returns mobility mode information back to the ePDG.

### SWd Access Authentication and Authorization

The SWd reference point is defined between a 3GPP AAA proxy and a 3GPP AAA server. The SWd interface is used in roaming scenarios where the 3GPP AAA proxy is located in the visited network and the 3GPP AAA server is located in the home network. The 3GPP AAA proxy acts as a Diameter proxy agent and forwards Diameter commands between the Diameter client and the Diameter server.

When used in connection with an STA reference point, the SWd interface supports the trusted non-3GPP access authentication and authorization procedure. For this procedure, the 3GPP AAA proxy forwards the Diameter commands received from the 3GPP AAA server and the trusted non-3GPP access network as a stateful Diameter proxy.

When used in connection with the SWm reference point, the SWd interface supports the untrusted non-3GPP access authentication and authorization procedure. For this procedure, the 3GPP AAA proxy forwards the Diameter commands received from the 3GPP AAA server and the ePDG as a stateful Diameter proxy.
SWx Authentication Procedure

The SWx is a reference point defined between a HSS and a 3GPP AAA server. It is used in AAA server registration of a new user.

The authentication procedure includes the following steps:

1. The 3GPP AAA server registers the current 3GPP AAA server address in the HSS for a given user when a new subscriber has been authenticated by the 3GPP AAA server.
2. The 3GPP AAA server informs the HSS about the current PDN GW identity and APN being used for a given UE, or that a certain PDN GW and APN pair is no longer used.
3. Accordingly, the 3GPP AAA server may de-register the currently registered 3GPP AAA server in the HSS for a given user and purge any related non-3GPP user status data in the HSS. This occurs if the UE for some reason has been disconnected from the non-3GPP access.

HSS Initiated Update of User Profile

The subscriber profile management procedures over SWx include the subscriber profile push and the subscriber profile request. The SWx reference point enables the following:

- Indication to the 3GPP AAA server of change of non-3GPP subscriber profile within HSS.
- Activation and deactivation of the subscriber and equipment trace in the PDN GW.

This procedure is used between the 3GPP AAA Server and the HSS and is invoked by the HSS during the following circumstances:

- When the subscriber profile has been modified and needs to be sent to the 3GPP AAA Server. This may happen due to a modification in the HSS.
- To update the 3GPP AAA Server with the identity of a dynamically allocated PDN GW, which is included in the APN-Configuration AVP in the user profile as a result of the first PDN connection establishment associated with an APN over 3GPP access.

This procedure is mapped to the Diameter command codes Push-Profile-Request (PPR) and Push-Profile-Answer (PPA). An IMSI Range based mechanism is provided to select the HSS server. In the CLI, if the MultiplePeersPolicy is IMSIRangeBased, then the ranges are configured as a list and from them the HSS server is selected.

S6b Authentication and Authorization Procedure

The S6b reference point is defined between a PDN GW and a 3GPP AAA server (for non-roaming case, or roaming with home routed traffic to PDN GW in home network) and between a PDN GW and a 3GPP AAA proxy. The S6b interface protocol is based on Diameter. It uses the Diameter base protocol and also supports Diameter EAP application. The EAP methods EAP-AKA and EAP-AKA` are used.

The authentication and authorization procedure includes the following steps:

1. The S6b interface enables authentication and authorization between the UE and the 3GPP AAA server/proxy.
2. When the UE performs the DSMIPv6 initial attach, it runs an IKEv2 exchange with the PDN GW. In this exchange, EAP AKA is used for UE authentication over IKEv2. The PDN GW acts as an IKEv2 responder and an EAP pass-through authenticator for this authentication.
3. The S6b authentication and authorization procedure is invoked by the PDN GW after receiving an IKE_SA_AUTH message from the UE. The S6b reference point performs authentication based on reuse of the DER/DEA command set defined in Diameter EAP.

3GPP Call Flows

When Prime Access Registrar receives an authentication or authorization request from any of the access points, it sends the packet to the rules and policy engine for processing. The rules and policies are added in the configuration of Prime Access Registrar. Client, vendor, and server scripting points are provided in order to modify any AVPs in the packet or to decide upon the type of service that the packet requires. For example, if the service is Radius-to-Diameter translation, Prime Access Registrar performs the translation of Radius packet to Diameter packet and sends it to the remote server. The service also has scripting points that can be used to modify the incoming packets. Once authentication or authorization is done for the packet session management can be performed for the packet by storing the user sessions in a session cache, if the authentication or authorization is successful. The server, client, vendor, and service outgoing scripting points can be used to modify the response packet.

Figure 20-2 depicts the call flows implemented for 3GPP.

This topic contains the following sections:
- CLI for 3GPP Authorization, page 20-12
- CLI for 3GPP Reverse Authorization, page 20-12
Contextualized text:

**CLI for 3GPP Authorization**

Following is the CLI for 3GPP authorization service:

```
[ //localhost/Radius/Services/3gpp-authz-diameter]
Name = 3gpp-authz-diameter
Description = STa to SWx authz (update-gw, get-profile, push-profile from HSS, de-reg from HSS)
Type = 3gpp-authorization
Protocol = Diameter
IncomingScript =
OutgoingScript =
SessionManager =
DiameterProxyService =
```

```
[ //localhost/Radius/Services/3gpp-authz-radius]
Name = 3gpp-authz-radius
Description = STa to SWx authz (update-gw, get-profile, push-profile from HSS, de-reg from HSS)
Type = 3gpp-authorization
Protocol = Radius
SessionManager =
TranslationService =
```

```
[ //localhost/Radius/Services/3gpp-authz-radius]
Name = 3gpp-authz-radius
Description = STa to SWx authz (update-gw, get-profile, push-profile from HSS, de-reg from HSS)
Type = 3gpp-reverse-authorization
Protocol = Radius
PreRequestTranslationScript =
PostRequestTranslationScript =
PreResponseTranslationScript =
PostResponseTranslationScript =
EnvMapping/
ForwardMapping/
ReverseMapping/
ResponseMapping/
```

```
[ //localhost/Radius/Services/3gpp-authz]
Name = 3gpp-authz-radius
Description = STa to SWx authz (update-gw, get-profile, push-profile from HSS, de-reg from HSS)
Type = 3gpp-reverse-authorization
Protocol = Diameter
Incoming =
Outgoing =
SessionManager =
RequestMapping/
EnvMapping/
ResponseMapping/
```

**CLI for 3GPP Reverse Authorization**

3GPP reverse authorization is used during RADIUS to Diameter translation. You can set the corresponding parameter to TRUE during the RADIUS to Diameter conversion. In this case, the request command mapping must not be defined because a new diameter request is created from the radius request by the 3GPP reverse authorization service. For more information about RADIUS<->Diameter translations, see [Translation Framework for Diameter, page 8-19](#).

Following is the CLI for 3GPP reverse authorization service:
[ //localhost/Radius/Services/reverse ]
Name = reverse
Description =
Type = 3gpp-reverse-authorization
IncomingScript- =
OutgoingScript- =
SessionManager = cache
TranslationService = diatorad
ProxyService =

[ //localhost/Radius/Services/diatorad ]
Name = diatorad
Description =
Type = diameter-radius
ProxyServiceName = rad-proxy
PreRequestTranslationScript- =
PostRequestTranslationScript- =
PreResponseTranslationScript- =
PostResponseTranslationScript- =
RequestMapping/
CommandMappings/
PPR = Radius-Access-Request
RAR = Radius-CoA-Request
AVPMappings/
Auth-Session-State = Cisco-AVPair
user-name = user-name
AVPsToBeAdded/
EnvironmentMappings/
ResponseMapping/
ResultCodeMappings/
Radius-CoA-ACK = Diameter-Success
Radius-CoA-NAK = Diameter-Unable-To-Deliver
AVPMappings/
AVPsToBeAdded/
EnvironmentMappings/
Using LDAP

This chapter provides information about using Lightweight Directory Access Protocol (LDAP) with Cisco Prime Access Registrar (Prime Access Registrar) to access information directories. You can use Prime Access Registrar to authenticate and authorize access requests by querying user information through LDAP.

Note

Prime Access Registrar supports LDAP version 3 and LDAP version 2 directory servers.

This chapter contains the following sections:

- Configuring LDAP
- Analyzing LDAP Trace Logs
- Bind-Based Authentication for LDAP

Configuring LDAP

To use LDAP in Prime Access Registrar, use aregcmd to do the following:

1. Configuring the LDAP Service.
4. Saving Your Configuration.

After you issue the save command, Prime Access Registrar attempts to validate the configuration, checks for all required properties, and ensures there is no logic error. If the validation is successful, Prime Access Registrar saves the configuration to the MCD database. When Prime Access Registrar is reloaded, it shuts down any current LDAP connections and builds new connections for the configured LDAP remote servers.
Configuring the LDAP Service

You configure an LDAP service under /Radius/Services. When you define an LDAP service under /Radius/Services, you must set its type to LDAP.

```
[ //localhost/Radius/Services/AR-LDAP ]
  Name = AR-LDAP
  Description =
  Type = ldap
  IncomingScript =
  OutgoingScript =
  OutagePolicy = RejectAll
  OutageScript =
  MultipleServersPolicy = Failover
  RemoteServers/
```

Table 21-1 describes the LDAP service properties.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; inherited from the upper directory</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the service</td>
</tr>
<tr>
<td>Type</td>
<td>Must be set to LDAP for LDAP service</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; must be set to AcceptAll or Drop Packet, or defaults to RejectAll</td>
</tr>
<tr>
<td>OutageScript</td>
<td>Optional</td>
</tr>
<tr>
<td>MultipleServersPolicy</td>
<td>Required; must be set to RoundRobin or defaults to Failover.</td>
</tr>
<tr>
<td>RemoteServers</td>
<td>Required; list of one or more remote servers defined under /Radius/Services/LDAP/RemoteServers. These servers must be listed in order under /Radius/RemoteServers.</td>
</tr>
</tbody>
</table>

This section contains the following topics:

- MultipleServersPolicy
- RemoteServers

MultipleServersPolicy

Use the MultipleServersPolicy property to configure the LDAP remote servers in RoundRobin mode, or the default Failover mode applies. When set to Failover, Prime Access Registrar directs requests to the first server in the /Radius/Services/LDAP/RemoteServers list. If that server should fail or go offline, Prime Access Registrar redirects all requests to the next server in the list. The process continues until Prime Access Registrar locates an online server.

When set to RoundRobin, Prime Access Registrar directs each request to the next server in the RemoteServers list to share the resource load across all listed servers.
RemoteServers

Use the RemoteServers directory to list one or more remote servers to process access requests. The servers must also be listed in order under `/Radius/RemoteServers`.

The order of the RemoteServers list determines the sequence for directing access requests when `MultipleServersPolicy` is set to `RoundRobin` mode. The first server in the list receives all access requests when `MultipleServersPolicy` is set to `Failover` mode.

Configuring an LDAP RemoteServer


The `Name`, `Protocol`, `Port`, `HostName`, `BindName`, `BindPassword`, `SearchPath`, and `Filter` properties must be configured to use an LDAP remote server.

The `Name`, `Protocol`, `Port`, `HostName`, `SearchPath`, and `Filter` properties must be configured to enable Bind-Based Authentication.

Note

You can leave the `BindName`, `BindPassword`, `UserPasswordAttribute`, `PasswordEncryptionStyle` and `DNSLookupAndLDAPRebindInterval` properties blank when you configure the Bind-Based Authentication feature in Prime Access Registrar.

Table 21-2 describes the LDAP Remote Server properties.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required name you assign</td>
</tr>
<tr>
<td>Description</td>
<td>Optional description of the server</td>
</tr>
<tr>
<td>Protocol</td>
<td>Required and must be set to LDAP; no default value</td>
</tr>
<tr>
<td>Port</td>
<td>Required; port on which LDAP server listens, default is port 389.</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Required; default is 300000 (ms)</td>
</tr>
<tr>
<td>Timeout</td>
<td>Required; specifies length of time Prime Access Registrar waits for a</td>
</tr>
<tr>
<td></td>
<td>response from the LDAP server before noting the server as down; default is</td>
</tr>
<tr>
<td></td>
<td>15 (seconds)</td>
</tr>
<tr>
<td>HostName</td>
<td>Required; specifies the hostname, FQDN, or IP address of the LDAP server</td>
</tr>
<tr>
<td>BindName</td>
<td>Specifies the distinguished name (DN) in the LDAP server for Prime Access</td>
</tr>
<tr>
<td></td>
<td>Registrar to bind with the LDAP server</td>
</tr>
<tr>
<td>BindPassword</td>
<td>Specifies the password for the distinguished name</td>
</tr>
<tr>
<td>UseSSL</td>
<td>FALSE by default</td>
</tr>
</tbody>
</table>
### Table 21-2 LDAP Remote Server Properties (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SearchPath~</td>
<td>Specifies search base to the organization and domain; for example: o=cisco.com</td>
</tr>
<tr>
<td>Filter~</td>
<td>(uid=%s) by default</td>
</tr>
<tr>
<td>UserPasswordAttribute</td>
<td>Should be set to the attribute in the directory server which stores users’ passwords; default is userpassword</td>
</tr>
<tr>
<td>LimitOutstandingRequests</td>
<td>FALSE by default</td>
</tr>
<tr>
<td>MaxOutstandingRequests</td>
<td>Limits the number of requests to the LDAP server; used to throttle the request load when the LDAP server does not function well under high TPS rates (default is 0)</td>
</tr>
<tr>
<td>MaxReferrals</td>
<td>Limits the number of referrals Prime Access Registrar allows when working with LDAPv2 (default is 0)</td>
</tr>
<tr>
<td>ReferralAttribute</td>
<td>LDAP attribute that contains a referral for LDAPv2</td>
</tr>
<tr>
<td>ReferralFilter</td>
<td>Filter used when following a referral for LDAPv2</td>
</tr>
<tr>
<td>PasswordEncryptionStyle</td>
<td>Dynamic by default; must be set to one of the following depending on the algorithm used by the LDAP server to encrypt passwords: Dynamic, Crypt, None, SHA-1, SSHA-1. When set to Dynamic, Prime Access Registrar analyzes the password and detects the encryption algorithm used. None indicates that the LDAP server stores clear text passwords. Note: If CHAP authentication is used with LDAP backing store, passwords in LDAP must be stored as clear text.</td>
</tr>
<tr>
<td>EscapeSpecialCharInUserName</td>
<td>FALSE by default</td>
</tr>
<tr>
<td>DNSLookupAndLDAPRebindInterval</td>
<td>Specifies the timeout period after which the Prime Access Registrar server will attempt to resolve the LDAP hostname to IP address (DNS resolution); 0 by default</td>
</tr>
<tr>
<td>DataSourceConnections</td>
<td>Specifies the number of concurrent connections to the LDAP server. The default value is 8.</td>
</tr>
<tr>
<td>SearchScope</td>
<td>Specifies how deep to search within a search path; default is SubTree which indicates a search of the base object and the entire subtree of which the base object distinguished name is the highest object. Base indicates a search of the base object only. OneLevel indicates a search of objects immediately subordinate to the base object, but does not include the base object.</td>
</tr>
</tbody>
</table>
Configuring LDAP

This section contains the following topics:

- DNS Look Up and LDAP Rebind Interval
- LDAPToRadiusMappings
- LDAPToEnvironmentMappings
- LDAPToCheckItemMappings

**Table 21-2 LDAP Remote Server Properties (continued)**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDAPToRadiusMappings</td>
<td>Optional; a list of name/value pairs in which the name is the name of the <code>ldap</code> attribute to retrieve from the user record, and the value is the name of the RADIUS attribute to set to the value of the <code>ldap</code> attribute retrieved. For example, when the <strong>LDAPToRadiusMappings</strong> has the entry: <code>FramedIPAddress = Framed-IP-Address</code>, the RemoteServer retrieves the <code>FramedIPAddress</code> attribute from the <code>ldap</code> user entry for the specified user, uses the value returned, and sets the Response variable <code>Framed-IP-Address</code> to that value.</td>
</tr>
<tr>
<td>LDAPToEnvironmentMappings</td>
<td>Optional; a list of name/value pairs in which the name is the name of the <code>ldap</code> attribute to retrieve from the user record, and the value is the name of the Environment variable to set to the value of the <code>ldap</code> attribute retrieved. For example, when the <strong>LDAPToEnvironmentMappings</strong> has the entry: <code>group = User-Group</code>, the RemoteServer retrieves the <code>group</code> attribute from the <code>ldap</code> user entry for the specified user, uses the value returned, and sets the Environment variable <code>User-Group</code> to that value.</td>
</tr>
<tr>
<td>LDAPToCheckItemMappings</td>
<td>Optional; a list of LDAP attribute/value pairs which must be present in the RADIUS access request and must match, both name and value, for the check to pass. For example, when the <strong>LDAPToCheckItemMappings</strong> has the entry: <code>group = User-Group</code>, the Access Request must contain the attribute <code>group</code>, and it must be set to <code>User-Group</code>.</td>
</tr>
<tr>
<td>UseBindBasedAuthentication</td>
<td>A boolean field that enables bind-based authentication with LDAP server. By default, this property is set to FALSE that uses existing legacy authentication method. On setting this property to TRUE, the mappings LDAPToRadius, LDAPToEnvironment, and LDAPToCheckItem will not work.</td>
</tr>
<tr>
<td>UseBinaryPasswordComparison</td>
<td>A boolean value that enables binary-based password comparison to authenticate. This property when set to TRUE, enables binary password comparison. By default this property is set to FALSE.</td>
</tr>
</tbody>
</table>

This section contains the following topics:

- DNS Look Up and LDAP Rebind Interval
- LDAPToRadiusMappings
- LDAPToEnvironmentMappings
- LDAPToCheckItemMappings
DNS Look Up and LDAP Rebind Interval

Prime Access Registrar provides a DNS Look-up and LDAP Rebind feature that enables you to use a smart DNS server for LDAP hostname resolution, allows you to query a DNS server at set intervals to resolve the LDAP hostname, and optionally rebind to the LDAP server, if necessary.

When you configure Prime Access Registrar to use an LDAP directory server, you can specify the hostname of the LDAP directory server. The hostname can be a qualified or an unqualified name. You can also specify a timeout period after which Prime Access Registrar will again resolve the hostname. If the IP address returned is different from the previous, Prime Access Registrar establishes a new LDAP bind connection.

The DNSLookupAndLDAPRebindInterval property specifies the timeout period after which the Prime Access Registrar server will attempt to resolve the LDAP hostname to IP address (DNS resolution). When you do not modify DNSLookupAndLDAPRebindInterval, the default value zero indicates the server will perform normal connection and binding only at start-up time or during a reload. Unless you change the default to a value greater than zero, the server will not perform periodic DNS lookups.

Prime Access Registrar maintains and uses the existing bind connection until a new one is established to minimize any performance impact during the transfer. Prime Access Registrar ensures that no requests are dropped or lost during the transfer to a new LDAP binding.

Set the DNSLookupAndLDAPRebindInterval using a numerical value and the letter H for hours or M for minutes, such as in the following examples:

\[
\text{set DNSLookupAndLDAPRebindInterval 15M—performs DNS resolution every 15 minutes}
\]

**Note**  
We recommend that you do not set DNSLookupAndLDAPRebindInterval to a value less than 15 minutes to minimize its effect on server performance.

\[
\text{set DNSLookupAndLDAPRebindInterval 1h—performs DNS resolution every hour}
\]

Configure the DNS Look-up and LDAP Rebind

To configure the DNS Look-up and LDAP Rebind,

**Step 1**  
Log into the Prime Access Registrar server, and use aregcmd to navigate to //localhost/Radius/Remoteservers. If necessary, add the LDAP server, or change directory to it.

\[
\text{cd /Radius/RemoteServers/ldap-serv1/}
\]

**Step 2**  
Set the DNSLookupAndLDAPRebindInterval property to the interval time desired.

\[
\text{set DNSLookupAndLDAPRebindInterval 30 M}
\]

LDAP Rebind Failures

Prime Access Registrar records any name resolution failures, bind successes and failures, and the destination hostname and IP address in the log file. At trace level 3, Prime Access Registrar also logs the time of any new bind connections and the closing of any old bind connections.
If either the name resolution or bind attempt fail, Prime Access Registrar continues using the existing bind connection until the timeout has expired again. If there is no existing bind connection, Prime Access Registrar marks the remote server object as *down*.

### LDAPToRadiusMappings

Configure LDAPToRadiusMappings with a list of *name/value* pairs where name is the name of the data store attribute to retrieve from the user record and the value is the name of the RADIUS attribute to set to the value of the data store attribute retrieved.

Values stored in a multivalued field in the LDAP directory are mapped to multiple RADIUS attributes. For example, if the LDAPToRadiusMappings has the following entry:

```
tunnel-info = Cisco-AVPair
```

The following LDAP fields in the user's record will create four Cisco-AVPair attributes in the user's Access-Accept RADIUS packet:

```
tunnel-info: vpdn:tunnel-id:ssg001
tunnel-info: vpdn:tunnel-type:12tp
tunnel-info: vpdn:ip-addresses=10.2.2.2
tunnel-info: vpdn:12tp-tunnel-password:secret
```

### LDAPToEnvironmentMappings

LDAPToEnvironmentMappings comprises a list of attribute name/value pairs or AV pairs where the name is the name of the data store attribute to retrieve from the user record, and the value is the name of the Environment variable to set to the value of the LDAP attribute retrieved.

For example, when the LDAPToEnvironmentMappings has the entry: `group =User-Group`, the RemoteServer retrieves the attribute from the LDAP user entry for the specified user, uses the value returned, and sets the Environment variable `User-Group` to that value.

### LDAPToCheckItemMappings

LDAPToCheckItemMappings comprises a list of LDAP AV pairs which must be present in the RADIUS access request and must match, both name and value, for the check to pass. Prime Access Registrar will first authenticate the user's password in the Access-Request before validating the check item attributes.

### Setting LDAP As Authentication and Authorization Service

Use `aregcmd` to configure the LDAP Service as the default authentication and authorization service under `/Radius` as in the following:

```
set DefaultAuthenticationService AR-LDAP
set DefaultAuthorizationService AR-LDAP
```

### Saving Your Configuration

When you use `aregcmd` to save your configuration, Prime Access Registrar does the following:

- Attempts to validate the configuration
Checks for all required parameters
Ensures there are no logic errors

If the validation is successful, Prime Access Registrar saves the configuration to the MCD database. When you `reload`, Prime Access Registrar shuts down any current LDAP connections and builds new connections for the configured LDAP servers.

This section contains the following topics:
- CHAP Interoperability with LDAP
- Allowing Special Characters in LDAP Usernames
- Dynamic LDAP Search Base

### CHAP Interoperability with LDAP

If you plan to use CHAP authentication with an LDAP backing store, the password in LDAP must be stored as clear text. This is due to the one-way hash used by the CHAP, crypt, SHA-1, and SSHA encryption algorithms.

### Allowing Special Characters in LDAP Usernames

This feature allows you to use special characters in LDAP usernames. The allowable special characters are *, (,), and \. These special characters can be included in the string passed to LDAP as the LDAP username value (usually the RADIUS username attribute).

The default of EscapeSpecialCharInUserName is FALSE. To enable this feature, use `aregcmd` to set the EscapeSpecialCharInUserName attribute in `/Radius/RemoteServers/ldap-server` to TRUE, as shown in the following example.

```bash
cd /Radius/RemoteServers/ldap-server
set EscapeSpecialCharInUserName  TRUE
```

```
/Radius/RemoteServers/Ldap-Server
EscapeSpecialCharinUserName  = TRUE
```

*Note*

This feature supports the LDAP V3 library.

### Dynamic LDAP Search Base

A new environment variable, Dynamic-Search-Path (see `rex.h`), can be used to set the dynamic LDAP search base. If this environment variable is defined for an LDAP service, it will override the default LDAP search base defined in the LDAP Remote Server configuration. This allows the LDAP search base to be configured on a per-user basis.

For example, you could match the search base to the organization and domain (in a Tcl script called from `/Radius/IncomingScript`):

```tcl
set user [ $request get User-Name ]
if { [ regexp {^[^@]+@([\^\.]+)(\.[\^\.]+)+$} $user m org domain ] } {
    $environ put Dynamic-Search-Path "ou=$org,ou=people,o=$domain"
}
Analyzing LDAP Trace Logs

Prime Access Registrar records in the log files any name resolution failures, bind successes and failures, and the destination hostname and IP address. At trace level 3, Prime Access Registrar logs the time of any new bind connections and the closure of any old bind connections and also information about user login requests and reply messages.

This section contains the following topics:

- Successful Bind Message
- Bind Failure Messages
- Login Failure Messages

Successful Bind Message

The following message is logged in the `name_radius_1_trace` file, when the Prime Access Registrar server successfully binds to the LDAP server. In this case, spatula-u5 is the LDAP server listening on port number 389.

10/12/2013 11:02:57: Log: Successfully bind to LDAP Server ldaps://spatula-u5:389

Bind Failure Messages

The following messages are logged in the `name_radius_1_trace` file, when Prime Access Registrar server fails to bind to the LDAP server.

10/12/2013 11:10:50: Log: Write in LDAPClient returned an error (32)
10/12/2013 11:10:50: Log: Remote LDAP Server ldaps://spatula-u5:387: Failed to open the connection to the LDAP server

Messages like those above could indicate that the hostname specified does not resolve to the correct IP address of the LDAP server or the configured port number might not be the port on which the LDAP server listens.

The following messages are logged in the `name_radius_1_trace` file, when Prime Access Registrar server fails to bind to the LDAP server.

10/12/2013 11:45:14: Log: Remote LDAP Server ldaps://spatula-u5:389: Unable to bind to LDAP Server: No such object ()
10/12/2013 11:45:14: Log: Remote LDAP Server ldaps://spatula-u5:389: Failed to open the connection to the LDAP server

The Distinguished Name (DN) provided in the BindName property was invalid. The DN provided in the BindName property should contain the exact string used in the directory server to define the object.

The following messages are logged in the `name_radius_1_trace` file, when Prime Access Registrar server fails to bind to the LDAP server.

10/12/2013 11:45:14: Log: Remote LDAP Server ldaps://spatula-u5:389: Unable to bind to LDAP Server: No such object ()
10/12/2013 11:45:14: Log: Remote LDAP Server ldaps://spatula-u5:389: Failed to open the connection to the LDAP server
10/12/2013 11:51:55: Log: Remote LDAP Server ldapserver (spatula-u5:389): Failed to open the connection to the LDAP server

The messages above indicate that the password provided in the BindPassword property was incorrect.

### Login Failure Messages

The following messages are logged in the `name_radius_1_trace` file, when user *jane* tries to login. These messages indicate that user *jane* does not have a record in the directory server or the SearchPath property has an incorrect value. The SearchPath property should have the directory where the user record is stored in the directory server.

Notice how the messages specify the service, remote LDAP server, username, and contents of the Access-Reject packet.

10/12/2013 11:24:17: P8457: Authenticating and Authorizing with Service AR-LDAP
10/12/2013 11:24:17: id = 5
10/12/2013 11:24:17: P8457: User jane was not found in the LDAP store
10/12/2013 11:24:17: P8457: Rejecting request
10/12/2013 11:24:17: P8457: Rejecting request
10/12/2013 11:24:17: P8457: Trace of Access-Reject packet
10/12/2013 11:24:17: P8457: identifier = 4
10/12/2013 11:24:17: P8457: length = 35

The following messages are logged in the `name_radius_1_trace` file, when user *bob* tries to login. These messages indicate the user record for user *bob* does not contain an attribute called pass. The UserPasswordAttribute property has an incorrect value called *pass*. The UserPasswordAttribute property should have the attribute name in the directory records where the user password is stored.

10/12/2013 11:36:59: P8461: Authenticating and Authorizing with Service AR-LDAP
10/12/2013 11:36:59: id = 7
10/12/2013 11:36:59: P8461: User bob's password does not match
10/12/2013 11:36:59: P8461: Rejecting request
10/12/2013 11:36:59: P8461: Rejecting request
10/12/2013 11:36:59: P8461: identifier = 6
10/12/2013 11:36:59: P8461: length = 35

The following messages are logged in the `name_radius_1_trace` file, when user *bob* tries to login. These messages indicate the user record for user *bob* does not contain an attribute called pass. The UserPasswordAttribute property has an incorrect value called *pass*. The UserPasswordAttribute property should have the attribute name in the directory records where the user password is stored.

10/12/2013 12:02:09: P9865: Authenticating and Authorizing with Service AR-LDAP
10/12/2013 12:02:09: id = 2
Bind-Based Authentication for LDAP

Prime Access Registrar supports most of the LDAP servers. But, a few of the LDAP servers do not support the functionality of Prime Access Registrar, which gets the passwords from the LDAP and matches them in Prime Access Registrar.

The bind-based authentication feature in Prime Access Registrar allows you to use any LDAP server; it verifies the password in the LDAP database instead of the Prime Access Registrar database. When Prime Access Registrar receives a request, it sends the username and password to the LDAP server. The LDAP server searches for a match, and approves the request if it finds a matching user credential in its database. It rejects the request if it does not find any matching credentials.

Configuring Bind-Based Authentication for LDAP

To configure the bind-based authentication for LDAP,

**Step 1** Launch `aregcmd`.

**Step 2** Create an LDAP service.

```bash
[ //localhost/Radius ]
   cd Services/
   add ldap
   cd ldap
   set Type ldap

[ //localhost/Radius/Services/ldap ]
   Name = ldap
   Description =
   Type = ldap
   IncomingScript~ =
   OutgoingScript~ =
   OutagePolicy~ = RejectAll
   OutageScript~ =
   MultipleServersPolicy = Failover
   RemoteServers/
```

Note: Remember to **reload** the Prime Access Registrar server after any changes to the LDAP server configuration.
cd RemoteServers
add 1 ldapserver

Step 3 Create the LDAP Remote Server Object.

[ //localhost/Radius ]
  cd RemoteServers
  add ldapserver
  cd ldapserver

[ //localhost/Radius/RemoteServers/ldap ]
  set Port <remote ldap server port number>
  set HostName <remote ldap server name/IP address>
  set SearchPath <configured in ldap server>
  set UseBindBasedAuthentication TRUE
  cd /Radius
  set DefaultAuthenticationService <ldap service name>
  set DefaultAuthorizationService <ldap service name>

Step 4 Save the configuration.
  save

Step 5 Restart the application.
  reload
Cisco Prime Access Registrar (Prime Access Registrar) supports Open Database Connectivity (ODBC) and Oracle Call Interface (OCI), open specifications that provide application developers a vendor-independent API with which to access data sources. For ODBC, Prime Access Registrar supports MySQL database connectivity and for OCI, it supports Oracle database connectivity. It provides RemoteServer objects and services to support ODBC or OCI. You can use Prime Access Registrar to authenticate and authorize access requests by querying user information through ODBC or OCI.

ODBC or OCI is an application program interface (API). Real data exchange between an application and data store is still carried out by SQL through ODBC or OCI. To achieve the most flexibility, you are required to define your own SQL using `aregcmd`. Prime Access Registrar will register the SQL statements and send them to the data store through ODBC or OCI when required. Because you can define your own SQL, Prime Access Registrar supports sites that have their own data stores.

ODBC is configured using `.ini` files, specifically `odbc.ini` and `odbcinst.ini`. However, you cannot create or modify these files directly. Prime Access Registrar creates the `.ini` files after you use `aregcmd` to configure the ODBC connection. The SQL is stored in the local database (MCD). During execution, the Prime Access Registrar server reads the local database, prepares the SQL statements, and sends the SQL to the data source.

**Note**

For OCI, the `.ini` files are not needed to connect to the database.

**Note**

Prime Access Registrar uses its own ODBC driver manager and does not share existing ODBC drivers (if you already have ODBC installed). If you are already using ODBC, you will have to maintain two separate ODBC installations.

The ODBC or OCI memory requirement depends on your configuration. The more datasources you configure, the more memory is required. Packet processing time might increase if you configure a large number of SQL statements under SQLDefinition.

The Prime Access Registrar package includes some ODBC and OCIIlib Drivers, and you should use the included driver whenever possible. If a data store’s ODBC driver is not included with Prime Access Registrar, you are required to install it. You configure the driver library using `aregcmd` to modify the associated `.ini` file.
This chapter contains the following sections:

- Oracle Software Requirements
- Configuring ODBC/OCI
- MySQL Support

**Oracle Software Requirements**

The Prime Access Registrar ODBC feature requires that you have MySQL driver packages. The OCI feature requires that you have Oracle client software installed. Supported Oracle client versions are 10.2.0.1.0 - 12c. All Oracle client software library files are expected under `$ORACLE_HOME/lib`.

When you install Prime Access Registrar software, the installation process prompts you for ORACLE_HOME variable and sets it in the Prime Access Registrar start-up script, `/etc/init.d/arserver`. Two other environment variables (ODBCINI and ODBCSYSINI) are also set in the `arserver` script. To change any of these variables, modify the `/etc/init.d/arserver` script and restart the Prime Access Registrar server.

**Note**

For OCI services, ensure that you have installed the Oracle client properly by using `tnsping` or `sqlplus` utilities. Oracle Instant Client libraries are not supported by OCI services.

**Configuring ODBC/OCI**

You use `aregcmd` to define your ODBC configuration and SQL statements. The Prime Access Registrar server automatically creates the `ODBC.ini` file for your driver manager and driver based on how you configure ODBC.

**Configuring the ODBC and ODBC-Accounting Remote Servers**

To use ODBC in Prime Access Registrar for AA:

- **Step 1** Configure an ODBC DataSource.
- **Step 2** Configure an ODBC RemoteServer object with protocol type as ‘odbc’.
- **Step 3** Configure an ODBC Service with service type as ‘odbc’.
- **Step 4** Set ODBC service as the DefaultAuthenticationService and DefaultAuthorizationService.
- **Step 5** Save your configuration.

To use ODBC in Prime Access Registrar for Accounting:

- **Step 1** Configure an ODBC DataSource.
- **Step 2** Configure an ODBC RemoteServer object with protocol type as ‘odbc-account’.
- **Step 3** Configure an ODBC Service with service type as ‘odbc-accounting’.
- **Step 4** Set ODBC service as the DefaultAccountingService.
Step 5  Save your configuration.

After you save and validate your configuration, it is saved in the MCD database. If you have configured an ODBC service, Prime Access Registrar will query the MCD database and create or modify the `odbc.ini` file before it builds a connection to the database. When you reload your configuration, Prime Access Registrar shuts down any existing ODBC connections, then queries the MCD database to create or modify the `odbc.ini` file and build a new connection for any configured ODBC Data Sources.

The following shows an example configuration for AA remote server:

```
[ //localhost/Radius/RemoteServers/oracle-access ]
Name = oracle-access
Description =
Protocol = odbc
ReactivateTimerInterval = 300000
Timeout = 15
DataSourceConnections = 8
ODBCDataSource = gordon
SNMPTrapIP =
SNMPTrapPort = 1521
KeepAliveTimerInterval = 0
SQLDefinition/
UserPasswordAttribute = password
SQLStatements/
Entries 1 to 1 from 1 total entries
Current filter: <all>

sql1/
   Name = sql1
   Description =
   Type = query
   SQL = "select password , username from arusers where username = ?"
   ExecutionSequenceNumber = 1
   MarkerList = UserName/SQL_CHAR
   ODBCToRadiusMappings/
   ODBCToEnvironmentMappings/
   ODBCToCheckItemMappings/
```

The following shows an example configuration for AAA remote server:

```
[ //localhost/Radius/RemoteServers/ora_acc ]
Name = ora_acc
Description =
Protocol = odbc-accounting
ReactivateTimerInterval = 1000
Timeout = 15
DataSourceConnections = 8
ODBCDataSource = gordon
SNMPTrapIP =
SNMPTrapPort = 1521
KeepAliveTimerInterval = 1000
BufferAccountingPackets = TRUE
MaximumBufferSize = "10 Megabytes"
NumberOfRetriesForBufferedPacket = 3
BackingStoreEnvironmentVariables =
UseLocalTimeZone = FALSE
AttributeList =
Delimiter =
SQLDefinition/
SQLStatements/
Entries 1 to 1 from 1 total entries
Current filter: <all>
```
Configuring ODBC/OCI

Configuring an ODBC/OCI Configuration

**You use aregcmd to define your ODBC configuration and SQL statements.**

**Configuring an OCI and OCI-Accounting Remote Servers**

To use OCI in Prime Access Registrar for AA:

**Step 1** Configure the DataSource type as oracle_oci.

**Step 2** Configure an OCI RemoteServer object protocol type as ‘oci’.

**Step 3** Configure an OCI Service with type as ‘oci’.

**Step 4** Set OCI service as the DefaultAuthenticationService and DefaultAuthorizationService.

**Step 5** Save your configuration.

To use OCI in Prime Access Registrar for Accounting:

**Step 1** Configure the DataSource type as oracle_oci.

**Step 2** Configure an OCI RemoteServer object protocol type as ‘oci-accounting’.

**Step 3** Configure an OCI Service with type as ‘oci-accounting’.

**Step 4** Set OCI service as the DefaultAccountingService.

**Step 5** Save your configuration.

After you save and validate your configuration, it is saved in the MCD database.

The following shows an example configuration for AA remote server:

```
sql/
Name = sql
Description =
Type = insert
SQL = 'insert into accounting(username,acct_status_type) values (?, ?)'
ExecutionSequenceNumber = 1
MarkerList = "UserName/SQL_CHAR Acct-Status-Type/SQL_CHAR"
```

You use aregcmd to define your OCI configuration and SQL statements.

**Configuring an OIC and OCI-Accounting Remote Servers**

To use OCI in Prime Access Registrar for AA:

**Step 1** Configure the DataSource type as oracle_oci.

**Step 2** Configure an OCI RemoteServer object protocol type as ‘oci’.

**Step 3** Configure an OCI Service with type as ‘oci’.

**Step 4** Set OCI service as the DefaultAuthenticationService and DefaultAuthorizationService.

**Step 5** Save your configuration.

To use OCI in Prime Access Registrar for Accounting:

**Step 1** Configure the DataSource type as oracle_oci.

**Step 2** Configure an OCI RemoteServer object protocol type as ‘oci-accounting’.

**Step 3** Configure an OCI Service with type as ‘oci-accounting’.

**Step 4** Set OCI service as the DefaultAccountingService.

**Step 5** Save your configuration.

After you save and validate your configuration, it is saved in the MCD database.

The following shows an example configuration for AA remote server:

```
[ //localhost/Radius/RemoteServers/oracle-access ]
Name = oracle-access
Description =
Protocol = oci
ReactivateTimerInterval = 300000
Timeout = 15
DataSourceConnections = 8
ODBCDataSource = gordon
SNMPTrapIP =
SNMPTrapPort = 1521
KeepAliveTimerInterval = 0
SQLDefinition/
UserPasswordAttribute = password
SQLStatements/
Enteries 1 to 1 from 1 total entries
Current filter: <all>

sql1/
Name = sql1
```
Configuring ODBC/OCI

You configure an ODBC or OCI service under /Radius/Services. When you define an ODBC or OCI service under /Radius/Services, you must set its type to ODBC or OCI and provide the following configuration options:

- **Configuring an ODBC/OCI Service**
- **Configuring an ODBC/OCI RemoteServer**
- **Configuring an ODBC DataSource**
- **Setting ODBC/OCI As Authentication and Authorization Service**
- **Setting ODBC/OCI As Accounting Service**
- **Saving Your Configuration**
- **Oracle Stored Procedures**
We will use ODBC or OCI as the ODBC or OCI service name in the following examples.

Example configuration for ODBC

```
[ //localhost/RADIUS/Services/ODBC ]
  Name = ODBC
  Description =
  Type = odbc
  IncomingScript =
  OutgoingScript =
  OutagePolicy = RejectAll
  OutageScript =
  MultipleServersPolicy = Failover
  RemoteServers/
```

Example configuration for OCI

```
[ //localhost/RADIUS/Services/OCI ]
  Name = OCI
  Description =
  Type = oci
  IncomingScript =
  OutgoingScript =
  OutagePolicy = RejectAll
  OutageScript =
  MultipleServersPolicy = Failover
  RemoteServers/
```

Table 22-1 describes the ODBC or OCI service parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; inherited from the upper directory</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the service</td>
</tr>
<tr>
<td>Type</td>
<td>Must be set to ODBC for ODBC service or OCI for OCI service</td>
</tr>
<tr>
<td>IncomingScript</td>
<td>Optional</td>
</tr>
<tr>
<td>OutgoingScript</td>
<td>Optional</td>
</tr>
<tr>
<td>OutagePolicy</td>
<td>Required; must be set to AcceptAll or Drop Packet, or defaults to RejectAll</td>
</tr>
<tr>
<td>OutageScript</td>
<td>Optional</td>
</tr>
<tr>
<td>MultipleServersPolicy</td>
<td>Required; must be set to RoundRobin or defaults to Failover.</td>
</tr>
<tr>
<td></td>
<td>When set to Failover, Prime Access Registrar directs requests to the first</td>
</tr>
<tr>
<td></td>
<td>server in the list until it determines the server is offline. If so,</td>
</tr>
<tr>
<td></td>
<td>Prime Access Registrar redirects all requests to the next server in the list</td>
</tr>
<tr>
<td></td>
<td>until it finds an online server.</td>
</tr>
<tr>
<td></td>
<td>When set to RoundRobin, Prime Access Registrar directs each request to the</td>
</tr>
<tr>
<td></td>
<td>next server in the RemoteServers list to share the resource load across all</td>
</tr>
<tr>
<td></td>
<td>servers in the RemoteServers list.</td>
</tr>
<tr>
<td>RemoteServers</td>
<td>Required list of remote servers defined under</td>
</tr>
<tr>
<td></td>
<td>/RADIUS/Services/ODBC/RemoteServers such as ODBC-Primary and ODBC-Secondary</td>
</tr>
</tbody>
</table>

Table 22-1 ODBC/OCI Service Parameters
Configuring an ODBC/OCI RemoteServer

Configuring an ODBC Remote Server

Configuring an OCI Remote Server

Table 22-2 describes the ODBC or OCI service parameters. The fields that are displayed in the table changes based on the protocol type selected.

Table 22-2  ODBC/OCI Remote Server Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; inherited from the upper directory</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the server</td>
</tr>
<tr>
<td>Protocol</td>
<td>Required and must be set to ODBC or OCI for ODBC or OCI service respectively; no default value</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Required; default is 300000 (ms)</td>
</tr>
<tr>
<td>Timeout</td>
<td>Required; default is 15 (seconds)</td>
</tr>
<tr>
<td>DataSourceConnections</td>
<td>Required; number of concurrent connections to data source (default is 8)</td>
</tr>
<tr>
<td>ODBCDataSource</td>
<td>Required; no default value</td>
</tr>
<tr>
<td>SQLDefinition</td>
<td>SQLDefinition/ (mandatory, no default); UserPasswordAttribute = (mandatory, no default; data store field for user password)</td>
</tr>
<tr>
<td></td>
<td>SQLStatements/</td>
</tr>
<tr>
<td></td>
<td>SQLStatement1/</td>
</tr>
<tr>
<td></td>
<td>SQLStatement2/</td>
</tr>
<tr>
<td>ODBCToRadiusMappings</td>
<td>Optional; a list of name/value pairs in which the name is the name of the odbc attribute to retrieve from the user record, and the value is the name of the RADIUS attribute to set to the value of the odbc attribute retrieved.</td>
</tr>
<tr>
<td></td>
<td>For example, when the ODBCToRadiusMappings has the entry: FramedIPAddress = Framed-IP-Address, the RemoteServer retrieves the FramedIPAddress attribute from the odbc user entry for the specified user, uses the value returned, and sets the Response variable Framed-IP-Address to that value.</td>
</tr>
<tr>
<td>Note</td>
<td>When you select the protocol as OCI, the field name will be displayed as OCToRadiusMappings.</td>
</tr>
</tbody>
</table>
Configuring ODBC/OCI

ODBCDataSource is the name of the datasource to be used by the remote server. An ODBCDataSource name can be reused by multiple remote servers. You configure ODBCDataSources under /Radius/Advanced/ODBCDataSources. See Configuring an ODBC Data Source, page 22-11, for more information.

### Tuning Parameters

1. SQLNET.OR.A timeout configuration

   Tuning $ORACLE_HOME/network/admin/sqlnet.ora file on the Oracle Client

   For proper function of the reactivate timer interval, one or more of the following parameters in sqlnet.ora file needs to be tuned:
   - SQLNET.INBOUND_CONNECT_TIMEOUT
   - SQLNET.SEND_TIMEOUT
   - SQLNET.RECV_TIMEOUT

   Ensure that the ReactivateTimerInterval of ODBC/ODBC-Accounting remoteservers is greater than the timeout values configured in sqlnet.ora.

2. AdditionalNativeOracleErrors connection lost error configuration

### ODBC/OCI Remote Server Parameters (continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ODBCToEnvironmentMappings (OCIToEnvironmentMappings)</td>
<td>Optional; a list of name/value pairs in which the name is the name of the odbc attribute to retrieve from the user record, and the value is the name of the Environment variable to set to the value of the odbc attribute retrieved. For example, when the ODBCToEnvironmentMappings has the entry: group = User-Group, the RemoteServer retrieves the group attribute from the odbc user entry for the specified user, uses the value returned, and sets the Environment variable User-Group to that value.</td>
</tr>
<tr>
<td><strong>Note</strong></td>
<td>When you select the protocol as OCI, the field name will be displayed as OCIToEnvironmentMappings.</td>
</tr>
<tr>
<td>ODBCToCheckItemMappings (OCIToCheckItemMappings)</td>
<td>Optional; a list of ODBC attribute/value pairs which must be present in the RADIUS access request and must match, both name and value, for the check to pass. For example, when the ODBCToCheckItemMappings has the entry: group = User-Group, the Access Request must contain the attribute group, and it must be set to User-Group.</td>
</tr>
<tr>
<td><strong>Note</strong></td>
<td>When you select the protocol as OCI, the field name will be displayed as OCIToCheckItemMappings.</td>
</tr>
</tbody>
</table>
Whenever OCI remote server oracle connection encounters configured ORA error, Prime Access Registrar will disconnect the remote server and reactivate it after the ReactivateTimerInterval.

Example

```plaintext
set /Radius/Advanced/AdditionalNativeOracleErrors
"3113,3114,12543,25408,25402,600,12502,12170,3135, 12518, 12526, 12528, 1089, 12547,
1041, 1092, 12537, 12514"
```

**SQL Definitions**

SQLDefinitions lists the UserPasswordAttribute and one or more SQL statements, listed numerically in the order to be run. The UserPasswordAttribute represents a column in the database that contains users’ password information. Individual SQLStatements are numbered SQL1 through SQLn under SQLStatements, as shown in the following example:

```plaintext
SQLDefinition/
  UserPasswordAttribute = asdfjkl
SQLStatements/
  SQL1/
  SQL2/
  SQL3/
  ...
```

The following example is an SQL statement used for Authentication and Authorization:

```plaintext
SQLStatements/
  SQL1
  Name = SQL1
  Type = query (mandatory, no default; must be query/procedure)
  SQL = SQL statement (mandatory, no default)
  ExecutionSequenceNumber = Sequence number for SQLStatement execution. (mandatory, no default and must be greater than zero).
  MarkerList = UserName/SQL_DATA_TYPE …… (mandatory, UserName must be defined)
```

For more information on stored procedures and stored functions, refer to Oracle Stored Procedures, page 22-13.

Table 22-3 describes the SQL Statement parameters.

**Table 22-3 SQL Statement Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name/number of SQL statement</td>
</tr>
<tr>
<td>Type</td>
<td>Query (mandatory, no default value)</td>
</tr>
<tr>
<td>SQL</td>
<td>SQL query statement</td>
</tr>
<tr>
<td>ExecutionSequenceNumber</td>
<td>Sequence number for SQLStatement execution, must be greater than zero (mandatory, no default)</td>
</tr>
<tr>
<td>MarkerList</td>
<td>Defines all markers for the query. MarkerList uses the format UserName/SQL_DATA_TYPE.</td>
</tr>
</tbody>
</table>
Chapter 22      Using Open Database Connectivity

Configuring ODBC/OCI

SQL Syntax Restrictions

You must observe the following SQL syntax restrictions in SQL queries for Prime Access Registrar.

1. The SQL statement must be in the format of `SELECT ... FROM ... WHERE ...` (Statements might be in lowercase.)

   Note  ‘WHERE’ is compulsory in the SQL statement.

2. Stored procedures with return value must be in the "begin ? := <Stored_procedure_name>(<IN/OUT Parameters>); end;" format.

3. Stored procedures without return value can be in the " CALL <Stored_procedure_name>(<IN/OUT Parameters>)" format.

4. Any arguments to Oracle functions like `distinct`, `count` must be given within braces, as shown in the following example:
   
   ```sql
   select distinct(attribute),password from profiles where username=?
   ```

   The resulted column from `distinct(attribute)` will be put into `attribute` which can be used for ODBC Mappings. The actual result set from Oracle for this column would be named `distinct(attribute)`.

5. The column list in the SQL statement must be delimited with a comma (,) and any extra spaces between statements are ignored. Aliasing for column names in SQL is not allowed. SQLDefinition properties define the SQL you want to execute, as shown in the following example.

Specifying More Than One Search Key

You can specify more than one search key for a table in the SQL SELECT. To do so, add another search criteria to the SQL statement and add the environment variable name to the MarkerList. For example, the following query and MarkerList can be used to look up a username and CLID match.

   ```sql
   select password from user_table where username = ? and clid = ?
   ```

   In this case, the marker list would look like this:

   ```sql
   UserName/SQL_CHAR clid/SQL_CHAR
   ```

   To configure the multiple entries in the MarkerList list, surround the entire string in double quotes like the following:

   ```sql
   set MarkerList "UserName/SQL_CHAR CLID/SQL_CHAR"
   ```

   To make this work, a variable called CLID must be in the environment dictionary. You can use a script to copy the appropriate value into the variable.

ODBCToRadiusMappings/OCIToRadiusMappings

You configure ODBCToRadiusMappings or OCIToRadiusMappings with a list of `name/value` pairs where name is the name of the data store attribute to retrieve from the user record and the value is the name of the RADIUS attribute to set to the value of the data store attribute retrieved.

For example, use the following `aregcmd` command to set a value for the variable `Framed-IP-Address`:

   ```bash
   set FramedIPAddress Framed-IP-Address
   ```
When the ODBCToRadiusMappings or OCIToRadiusMappings has this entry, the RemoteServer retrieves the attribute from the data store user entry for the specified user, uses the value returned, and sets the response variable `Framed-IP-Address` to that value.

When an SQL select statement returns more than one row for a column mapped under ODBCToRadiusMappings or OCIToRadiusMappings, multiple Radius attributes are created.

For example, consider the following SQL select statement with ciscoavpair configured to Cisco-AVPair under ODBCToRadiusMappings. The table.column syntax requires an SQL alias for the mapping to work, as shown in the following example:

```
SQLStatements/
 SQL1/  
   select table1.abc as t1abc, password from table2 where username = ?  
   Mapping: t1abc = my_mapping
```

If two rows are returned for ciscoavpair column, two Cisco-AVPair attributes will be created.

**ODBCToEnvironmentMappings/OCIToEnvironmentMappings**

Under ODBCToEnvironmentMappings or OCIToEnvironmentMappings there is a list of name and value pairs in which the name is the name of the data store attribute to retrieve from the user record, and the value is the name of the Environment variable to set to the value of the ODBC or OCI attribute retrieved.

For example, when the ODBCToEnvironmentMappings has the entry: group = User-Group, the RemoteServer retrieves the attribute from the ODBC user entry for the specified user, uses the value returned, and sets the environment variable User-Group to that value. When an SQL select statement returns more than one row for a column mapped under ODBCToEnvironmentMappings, the value for all rows is concatenated and assigned to the environment variable.

**ODBCToCheckItemMappings/OCIToCheckItemMappings**

A list of ODBC or OCI attribute/value pairs which must be present in the RADIUS access request and must match, both name and value, for the check to pass.

For example, when the ODBCToCheckItemMappings or OCIToCheckItemMappings has the entry: `group = User-Group`, the Access Request must contain the attribute `group`, and it must be set to `User-Group`.

**Configuring an ODBC DataSource**

ODBCDataSource is the name of the datasource to be used by the remote server. You configure ODBCDataSources under `/Radius/Advanced/ODBCDataSources`. Multiple remote servers can use the same ODBCDataSource.

Under the ODBCDataSource object definition, for ODBC a list defines `ODBC.ini` filename/value pairs for a connection. The list includes a Type field and a Driver field, different for each Driver and Data Source, to indicate its Driver and Data Source. Prime Access Registrar currently supports only the Easysoft Open Source Oracle Driver.

For OCI services, ODBCDataSource type should be `oracleoci`. The following is an example configuration of ODBCDataSource for OCI services.

```plaintext
[ //localhost/Radius/Advanced/ODBCDataSources/gordon ]
Name = gordon
Description =
Type = oracleoci
```
UserID = scott  
Password = <encrypted>  
DataBase = orcl.cisco.com

Table 22-4 describes the Easysoft Open Source Oracle Driver options for ODBC.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the ODBCDataSource</td>
</tr>
<tr>
<td>Type</td>
<td>Required; type of the ODBC data source, which could be myodbc or oracle_oci</td>
</tr>
<tr>
<td>Driver</td>
<td>Required; liboardbc.so (default value)</td>
</tr>
<tr>
<td>Database</td>
<td>Required; Oracle Client configuration database name (no default value)</td>
</tr>
<tr>
<td>UserID</td>
<td>Required; database username (no default value)</td>
</tr>
<tr>
<td>Password</td>
<td>Optional user password; shown encrypted</td>
</tr>
</tbody>
</table>

Table 22-5 describes the OCILib Open Source Oracle Driver options for OCI.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the ODBCDataSource</td>
</tr>
<tr>
<td>Type</td>
<td>Required; must be Oracle_oci</td>
</tr>
<tr>
<td>Database</td>
<td>Required; Oracle Client configuration database name (no default value)</td>
</tr>
<tr>
<td>UserID</td>
<td>Required; database username (no default value)</td>
</tr>
<tr>
<td>Password</td>
<td>Optional user password; shown encrypted</td>
</tr>
</tbody>
</table>

### Setting ODBC/OCI As Authentication and Authorization Service

Use `aregcmd` to configure the ODBC Service as the default authentication and authorization service under `//localhost/RADIUS` as in the following:

```
set DefaultAuthenticationService odbc-service
set DefaultAuthorizationService odbc-service
```

Use `aregcmd` to configure the OCI Service as the default authentication and authorization service under `//localhost/RADIUS` as in the following:

```
set DefaultAuthenticationService oci-service
set DefaultAuthorizationService oci-service
```
When you use an ODBC or OCI service, configure the BackingStoreDiscThreshold property under /Radius/Advanced to ensure that the data generated by log files do not exceed the size limit configured.

**Setting ODBC/OCI As Accounting Service**

Use `aregcmd` to configure the ODBC Service as the default accounting service under /localhost /Radius as in the following:

```
set DefaultAccountingService odbc-service
```

Use `aregcmd` to configure the OCI Service as the default authentication and authorization service under /localhost /Radius as in the following:

```
set set DefaultAccountingService oci-service
```

**Saving Your Configuration**

When you use `aregcmd` to save your configuration, Prime Access Registrar attempts to validate the configuration, checks for all required parameters, and ensures there is no logic error. If the validation is successful, the configuration is saved to the MCD database. When you **reload**, Prime Access Registrar shuts down any current ODBC/OCI connections and builds new connections for the configured ODBC Data Sources.

**Oracle Stored Procedures**

A stored procedure is a database procedure similar to other programming language procedures, which is contained within the database itself. A SQL Server stored procedure that contains one or more IN parameters are used to pass data into the stored procedure. Similarly, one or more OUT parameters in the stored procedure are used to return data back to the calling application. Prime Access Registrar supports Oracle stored procedures/functions with IN and OUT parameters only over the OCI interface.

For Authentication and Authorization, Prime Access Registrar supports both Stored Procedures and Stored Functions with the In/Out parameters and return value. In the configuration for the AA remote server, the UserPasswordAttribute value must be in the marker list for procedures.

For Accounting, Prime Access Registrar supports both Stored Procedures and Stored Functions with only the In parameters, and does not support return value and Out parameters.

The following are the examples for stored functions and procedures calling inside Prime Access Registrar:

**Example format for stored functions with return value**

```
SQL = "begin ? := stress (?);end;"
```

**Example for stored procedures**

```
SQL = " CALL Accounting_Request( ?,?,?,?)"
```
Prime Access Registrar does not support, return value with the "call" format for the stored procedures.

The following shows an example configuration for OCI AA remote server:

```plaintext
[ //localhost/Radius/RemoteServers ]
Entries 1 to 2 from 2 total entries
Current filter: <all>

oci-access/
   Name = oci-access
   Description =
   Protocol = oci
   ReactivateTimerInterval = 300000
   Timeout = 15
   DataSourceConnections = 8
   ODBCDataSource = 54
   SNMPTrapIP = 10.77.240.57
   SNMPTrapPort = 1521
   KeepAliveTimerInterval = 0
   SQLDefinition/
      UserPasswordAttribute = password
   SQLStatements/
      Entries 1 to 1 from 1 total entries
      Current filter: <all>

   sql1/
      Name = sql1
      Description =
      Type = procedure
      SQL = "begin ? := stress (?);end;"
      ExecutionSequenceNumber = 1
      MarkerList = "password/SQL_OUT UserName/SQL_CHAR"
   OCIToRadiusMappings/
   OCIToEnvironmentMappings/
   OCIToCheckItemMappings/
```

The following shows an example configuration for OCI AA remote server:

```plaintext
oci-acc/
   Name = oci-acc
   Description =
   Protocol = oci-accounting
   ReactivateTimerInterval = 300000
   Timeout = 15
   DataSourceConnections = 8
   ODBCDataSource = 54
   SNMPTrapIP =
   SNMPTrapPort = 1521
   KeepAliveTimerInterval = 0
   BufferAccountingPackets = TRUE
   MaximumBufferSize = "10 Megabytes"
   NumberOfRetriesForBufferedPacket = 3
   BackingStoreEnvironmentVariables =
   UseLocalTimeZone = FALSE
   AttributeList =
   Delimiter =
   SQLDefinition/
   UserPasswordAttribute =
   SQLStatements/
   Entries 1 to 1 from 1 total entries
   Current filter: <all>
```
MySQL Support

Prime Access Registrar provides support for MySQL to query user records from a MySQL database and enables you to write accounting records into MySQL when using Oracle accounting. Prime Access Registrar has been tested with MySQL 5.0.90 and MyODBC 3.51.27 (reentrant).

This section contains the following topics:

- MySQL Driver
- Configuring a MySQL Datasource
- Example Configuration

MySQL Driver

You can download the MySQL driver from the MySQL website at [http://mysql.com](http://mysql.com). You can go directly to the driver download page using the following URL:

```
```

Save the downloaded file to a temporary location such as `/tmp`. Use commands to unzip and install the driver.

For better performance with mysql, add the following code to the odbcinst.ini file under the `/cisco-ar/odbc/etc` directory:

```
[mysql]
DRIVER=/usr/lib/libmyodbc3_r.so
Threading = 1
```

`libmyodbc3_r.so` is the driver location

Configuring a MySQL Datasource

You require the following to configure a MYSQL Datasource:

- ODBCDataSource object
- RemoteServer object
- ODBC service
- Default AA services

Configuring a MYSQL datasource

To configure the Prime Access Registrar server to query records from a MySQL database:
**MySQL Support**

**Step 1**
Log into the Prime Access Registrar server and launch `aregcmd`.
Log in as a user with administrative rights such as user `admin`.

**Step 2**
Change directory to the `//localhost/Radius/Advanced/ODBCDataSources` and add a new ODBCDataSource.

```
cd //localhost/Radius/Advanced/ODBCDataSources
add mysql
```

**Step 3**
Set the new ODBCDatasource type to myodbc.

```
cd mysql
```

The following is the default configuration for an ODBCDataSource object of type myodbc:

```
[ //localhost/Radius/Advanced/ODBCDataSources/mysql ]
Name = mysql
Description =
Type =
```

```
set type myodbc
```

**Step 4**
Set the Driver property to the path of the MyODBC library. Use a command like the following:

```
set driver /scratch/myodbc/libmyodbc3_r.so
```

**Step 5**
Set the UserID property to a valid username for the MyODBC database and provide a valid password for this user.

```
set userid ar-mysql-user
set password biscuit
```

**Step 6**
Provide a DataBase name and the name of the Prime Access Registrar RemoteServer object to associate with the ODBCDataSource.

```
set database database_name
set server remote_server_name
```

**Step 7**
Change directory to `//Radius/RemoteServers` and add a RemoteServer object to associate with the new ODBCDataSource.

```
cd //Radius/RemoteServers
add mysql
```
**Step 8** Change directory to the new RemoteServer and set its protocol to odbc.

```
  cd mysql
  set protocol odbc
```

**Step 9** Set the ODBCDataSource property to the name of the ODBCDataSource to associate with this RemoteServer object.

```
  set ODBCDataSource mysql
```

**Step 10** Change directory to `/Radius/Services` and add an ODBC service as described in Configuring an ODBC/OCI Service, page 22-5.

**Step 11** Change directory to `/Radius` and set the DefaultAuthenticationService and DefaultAuthorizationService properties to the ODBC service added in the previous step.

---

## Example Configuration

The following shows an example configuration for a MySQL ODBC data source. See Configuring an ODBC Data Source, page 22-11 for more information.

```
[ //localhost/Radius/Advanced/ODBCDataSources/mysql ]
  Name = mysql
  Type = myodbc
  Driver = /tmp/libmyodbc3_r.so
  UserID = mysql
  Password = <encrypted>
  DataBase = test
  Server = mysql-a 
  Port = 3306
```

The following shows an example configuration for a RemoteServer. See Configuring an ODBC/OCI Remote Server, page 22-7 for more information.

```
[ //localhost/Radius/RemoteServers/mysql-a ]
  Name = mysql
  Description =
  Protocol = odbc
  ReactivateTimerInterval = 300000 
  Timeout = 15
  DataSourceConnections = 8
  ODBCDataSource = mysql
  KeepAliveTimerInterval = 0
  SQLDefinition/
  UserPasswordAttribute = asdfjkl
  SQLStatements/ 
      SQL1/
          Name = SQL1 
          Type = query (mandatory, no default; must be query)
          SQL = SQL statement (mandatory, no default)
          ExecutionSequenceNumber = Sequence number for SQLStatement execution.(mandatory, no default and must be greater than zero).
          MarkerList = UserName=SQL_DATA_TYPE …… (mandatory, UserName must be defined)
      SQL2/
      SQL3/
```

```
  ODBCToRadiusMappings/
  ODBCToEnvironmentMappings/
```
The following shows an example configuration for an ODBC service. See Configuring an ODBC/OCI Service, page 22-5 for more information.

```plaintext
[ //localhost/Radius/Services/ODBC ]
Name = ODBC
Description =
Type = ODBC
IncomingScript~ =
OutgoingScript~ =
OutagePolicy~ = RejectAll
OutageScript~ =
MultipleServersPolicy = Failover
RemoteServers/
  1. mysql-a
```

The following shows an example configuration where the DefaultAuthenticationService and DefaultAuthorizationService properties have been set to the ODBC service.

```plaintext
[ //localhost/Radius ]
Name = Radius
Description =
Version = 5.1
IncomingScript~ =
OutgoingScript~ =
DefaultAuthenticationService~ = ODBC
DefaultAuthorizationService~ = ODBC
```
SIGTRAN-M3UA

SIGTRAN, a working group of the Internet Engineering Task Force (IETF), has defined a protocol for the transport of real-time signaling data over IP networks. Cisco Prime Access Registrar (Prime Access Registrar) supports SS7 messaging over IP (SS7oIP) via SIGTRAN-M3UA, a new transport layer which leverages Stream Control Transmission Protocol (SCTP). Prime Access Registrar supports SIGTRAN-M3UA to fetch the authentication vectors from HLR, which is required for EAP-AKA/EAP-SIM authentication.

Note
You have SIGTRAN-M3UA interface support in addition to the existing SUA interface support.

The EAP-AKA and EAP-SIM authentication service is extended to use M3UA. When using M3UA service for authentication, the subscriber identity (IMSI) is used to send a request to HLR and receives information from HLR containing the authentication information for authenticating an user. The authentication service initiates a request to the SIGTRAN server using IMSI, which retrieves the configured number of authentication vectors from HLR, i.e Triplets or Quintets.

Figure 23-1  MAP Service

The Prime Access Registrar server initiates the MAP service. After enabling the MAP service, the Prime Access Registrar server sends a sendAuthenticationInfo request that contains IMSI and the number of requested authentication vectors to HLR. The HLR sends a response containing the requested vectors information to Prime Access Registrar. Next, the Prime Access Registrar server sends a sendRoutingInfoForLCS request that contains IMSI and the GMLC address to HLR. The HLR sends a response containing the MSISDN information for authenticating the mobile subscribers.

Prime Access Registrar provides map-restore-data authentication support for m3ua services. For more information, see Configuring M3UA Service with Map Restore Data Authorization, page 23-15.
Prime Access Registrar supports multiple remote servers with the protocol type, SIGTRAN-M3UA. However, Prime Access Registrar validates and ensures the following when multiple remote servers are available:

- The source port is different for all the remote servers.
- If Origin Point Code (OPC) is different, the routing context is also different for all the remote servers.
- The Destination Point Code (DPC) is different for all the remote servers.
- The NetworkVariant, SubServiceField (SSF), TCAPVariant, NetworkAppearance, and NetworkIndicator values are the same for all the remote servers.

This section describes the following:

- Prerequisites to SIGTRAN-M3UA
- Configuring EAP-AKA/EAP-SIM with SIGTRAN-M3UA
- Blacklisting IMSI Values, page 23-13
- Configuring M3UA Service
- Support for SCTP Multihoming in SIGTRAN-M3UA, page 23-22
- Tuning Global SIGTRAN Parameters, page 23-23
- SIGTRAN-M3UA Logs, page 23-25

Prerequisites to SIGTRAN-M3UA

Before enabling the SIGTRAN-M3UA remote server, you must do the following:

- ensure that LKSCTP is not available in the Prime Access Registrar server.
- ensure to restart the Prime Access Registrar server whenever you make any configuration changes.
- ensure that you have the 32-bit rpm files for the relevant RHEL OS versions while installing the Cisco Prime Access Registrar. For the list of required rpms for the relevant OS versions, see Required 32-bit rpms for Relevant RHEL OS Versions.

**Note**

You must install the rpm versions relevant to the RHEL OS versions while installing the Prime Access Registrar.

- ensure that the ‘bc’ command (which is an arbitrary precision calculator language) is present while installing Prime Access Registrar in a Linux machine. If the ‘bc’ command is not present, install the relevant rpm such as bc-1.06.95-1.el6.x86_64 on that machine.
- ensure that you have the following packages while installing the Prime Access Registrar:
  - gcc version-3.4.6
  - gdome-config-0.8.1

**Note**

You need to build the gdome-config-0.8.1 package to make it available. For more information, see Building gdome Package, page 23-3

- xml2-config-2.6.23
Prerequisites to SIGTRAN-M3UA

- pkg-config-0.15.0
- glib-2.30
- gtk-2.41
- libxml-2.2.6.20

Required 32-bit rpms for Relevant RHEL OS Versions

<table>
<thead>
<tr>
<th>rpm</th>
<th>RHEL OS Version 6.4</th>
<th>RHEL OS Version 6.6</th>
<th>RHEL OS Version 7.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>glibc</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>gdome2</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>glib</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>glib2</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>libgcc</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>libstdc++</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>libxml2</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ncurses</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>nspr</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>nss</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>zlib</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>nss-softokn-freebl</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ncurses-libs</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>nss-util</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>gamin</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>libselinux</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Building gdome Package

To build gdome-config-0.8.1 package:

Step 1  Download gdome2-0.8.1.tar.gz package from the location [http://gdome2.cs.unibo.it/#downloads](http://gdome2.cs.unibo.it/#downloads).

Step 2  Execute the following command:

```
gunzip gdome2-0.8.1.tar.gz
```

Step 3  Untar the package using the following command:

```
tar -xvf gdome2-0.8.1.tar
```

Step 4  Use the `cd` command to move into the package obtained from Step 3.

Step 5  Execute the following commands:

```
./configure --prefix=<GdomeInstallPath> --with-glib-prefix=<GlibInstalledDirectory>
maker
make install
```

Where,
• GdomeInstallPath specifies where the Gdome libraries must be placed.
• GlibInstalledDirectory specifies which directory the Glib libraries reside in the filesystem.

**Step 6**

Now gdome libraries will be available in the location GdomeInstallPath.

---

## Configuring EAP-AKA/EAP-SIM with SIGTRAN-M3UA

You can use aregcmd to create and configure the service of type eap-aka or eap-sim, see EAP-AKA or EAP-SIM for more information.

To configure EAP-AKA service with SIGTRAN-M3UA remote server:

**Step 1**

Launch aregcmd.

**Step 2**

Create an EAP-AKA service.

```
  cd /Radius/Services
  add eap-aka-service
```

**Step 3**

Set type as eap-aka.

```
  set eap-aka
```

**Step 4**

Add m3ua remote server in the remoteServers

```
  cd remoteServers/
  Set 1 m3ua
```

The following shows an example configuration for EAP-AKA service with SIGTRAN-M3UA remote server support, see Table 9-1 to know more about EAP-AKA service properties.

```
[ //localhost/Radius/Services ]
Entries 1 to 2 from 2 total entries
Current filter: <all>

eap-aka/
  Name = eap-aka
  Description =
  Type = eap-aka
  AlwaysRequestIdentity = False
  EnableIdentityPrivacy = False
  PseudonymSecret = <encrypted>
  PseudonymRenewtime = "24 Hours"
  PseudonymLifetime = Forever
  Generate3GPPCompliantPseudonym = False
  EnableReauthentication = False
  MaximumReauthentications = 16
  ReauthenticationTimeout = 3600
  ReauthenticationRealm =
  AuthenticationTimeout = 120
  QuintetGenerationScript~ =
```
UseProtectedResults = False
SendReAuthIDInAccept = False

Subscriber_DBLookup = SIGTRAN-M3UA
FetchAuthorizationInfo = FALSE
MultipleServersPolicy = Failover
IncomingScript- =
OutgoingScript- =
OutageScript- =
RemoteServers/

To configure EAP-SIM service with SIGTRAN-M3UA remote server:

---

**Step 1** Launch `aregcmd`.

**Step 2** Create an EAP-SIM service.

```bash
cd /Radius/Services
add eap-sim-service
```

**Step 3** Set type as `eap-sim`.

```bash
set eap-sim
```

**Step 4** Add m3ua remote server in the `remoteServers`

```bash
cd remoteServers
Set 1 m3ua
```

---

The following shows an example configuration for EAP-SIM service with SIGTRAN-M3UA remote server support. See Table 9-6 to know more about EAP-SIM service properties.

```bash
eap-sim/
   Name = eap-sim
   Description =
   Type = eap-sim
   NumberOfTriplets = 2
   UseSimDemoTriplets = False
   AlwaysRequestIdentity = False
   EnableIdentityPrivacy = False
   PseudonymSecret = "<encrypted>
   PseudonymRenewtime = "24 Hours"
   PseudonymLifetime = Forever
   Generate3GPPCompliantPseudonym = False
   EnableReauthentication = False
   MaximumReauthentication = 16
   ReauthenticationTimeout = 3600
   ReauthenticationRealm =
   TripletCacheTimeout = 0
   AuthenticationTimeout =
   UseProtectedResults = False
   SendReAuthIDInAccept = False
   SubscriberDBLookup = SIGTRAN-M3UA
   FetchAuthorizationInfo = FALSE
   MultipleServersPolicy = Failover
   IncomingScript- =
   OutgoingScript- =
   OutageScript- =
```
Configuring EAP-AKA/EAP-SIM with SIGTRAN-M3UA

RemoteServers/

Note
After enabling the SIGTRAN-M3UA remote server, you must ensure to restart the Prime Access Registrar server whenever you make any configuration changes.

Note
If you set FetchAuthorizationInfo as TRUE for EAP-AKA or EAP-SIM service for SIGTRAN-M3UA in Prime Access Registrar, it fetches the MSISDN information from HLR in response. The following is an example script for reading the MSISDN information from the response,

```bash
proc MapMSISDN {request response environ} {
    $environ get AuthorizationInfo
}
```

Configuring SIGTRAN-M3UA Remote Server
You can configure the SIGTRAN-M3UA remoteserver under /Radius/RemoteServers.

To configure the SIGTRAN-M3UA remote server:

**Step 1**
Launch aregcmd.

**Step 2**
Create sigtran-m3ua remote server.

```bash
cd /r/remoteServers/
add M3UA
cd M3UA
set protocol sigtran-m3ua
```

**Step 3**
Set the Subscriber_DBLookup.

```bash
set Subscriber_DBLookup SIGTRAN-M3UA
```

**Step 4**
Set the port of the HLR.

```bash
set DestinationPort 2905
```

**Step 5**
Set the port for the source.

```bash
set SourcePort 2905
```

**Step 6**
Set the reactivate timer interval for the remote server.

```bash
set the reactivate timer interval.
```

**Step 7**
Set the subsystem number for the local.

```bash
set LocalSubSystemNumber 149
```
Step 8  Set routingindicator.

   Set routingindicator rte_gt

Step 9  Set mlcnumber.

   Set mlcnumber

Step 10 Set routingparameters.

   cd routingparameters/
   set OriginPointCode 2
   set DestinationPointCode 4
   set RemoteSubSystemNumber 6
   set OPCMask 16383
   set DPCMask 16383
   set RoutingContext 11

Step 11 Set the source and destination gt parameters.

Step 12 Set the numbering plan, encoding scheme, format, and digits for source.

Step 13 Set the numbering plan, encoding scheme, format, and digits for destination.

ANSI Support for SIGTRAN

Prime Access Registrar provides ANSI variant support apart from ITU variants in SIGTRAN stack for EAP-SIM and EAP-AKA services to M3UA.

While using this service for authentication, the subscriber identity (IMSI) is obtained from the request. Using this IMSI, the authentication service initiates a request to the SIGTRAN server. This request is to retrieve the configured number of authentication vectors (triplets/quintets) for the IMSI.

The remote SIGTRAN server initiates the IS41 service primitive Authentication Data request with IMSI and number of requested authentication vectors. This will retrieve the authentication vectors from HLR which will be used by the authentication service for authenticating the mobile subscriber.

Note  Prime Access Registrar supports either ITU or ANSI variant in one running instance. Both the variants are not supported simultaneously.
The following shows an example configuration of SIGTRAN-M3UA remote server with ITU variant:

```
//localhost/Radius/RemoteServers/m3ua
Name = m3ua
Description = Protocol = sigtran-m3ua)
SourcePort = 2905
LocalSubSystemNumber = 149
DestinationPort = 2905
IMSITranslationScript~ = GlobalTitleTranslationScript~ = setGT
Timeout = 15
ReactivateTimerInterval = 2000
LimitOutstandingRequests = FALSE
MaxOutstandingRequests = 0
MaxRetries = 3
MAPVersion = 2
NetworkVariant = ITU
SubServiceField = NAT
TCAPVariant = ITU96
NetworkAppearance = 1
NetworkIndicator = NAT
MLCNumber = 123456789012345
TrafficMode = LOADSHARE
LoadShareMode = SLS
RoutingIndicator = RTE_GT
RoutingParameters/
  OriginPointCode = 2
  DestinationPointCode = 4
  RemoteSubSystemNumber = 6
  OPCMask = 16383
  DPCMask = 16383
  ServiceIndicatorOctet = 0
  RoutingContext = 11
SourceGTAddress/
  SourceGTDigits = 919845071842
  SourceGTFormat = GTFRMT_4
  SourceNatureofAddress = INTNUM
  SourceTranslationType = 0
  SourceNumberingPlan = ISDN
  SourceEncodingScheme = BCDEVEN
DestinationGTAddress/
  DestGTDigits = 919845071842
  DestGTFormat = GTFRMT_4
  DestNatureofAddress = INTNUM
  DestTranslationType = 0
  DestNumberingPlan = ISDN
  DestEncodingScheme = BCDEVEN
```

Table 23-1 describes SIGTRAN-M3UA remote server properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Required; inherited from the upper directory.</td>
</tr>
<tr>
<td>Description</td>
<td>An optional description of the service.</td>
</tr>
<tr>
<td>Protocol</td>
<td>Represents the type of remote server. The value should be SIG-TRAN-M3UA.</td>
</tr>
<tr>
<td>SourcePort</td>
<td>The port number in which Prime Access Registrar is installed for M3UA transactions.</td>
</tr>
</tbody>
</table>
### Table 23-1 SIGTRAN-M3UA Stack Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LocalSubSystemNumber</td>
<td>The local sub system number is set as 149 by default.</td>
</tr>
<tr>
<td>DestinationPort</td>
<td>The destination port number to which Prime Access Registrar connects.</td>
</tr>
<tr>
<td>IMSITranslationScript</td>
<td>The scripting point is used to modify the IMSI based on the requirement</td>
</tr>
<tr>
<td></td>
<td>before sending the request to STP/HLR.</td>
</tr>
<tr>
<td>GlobalTitleTranslationScript</td>
<td>This is used to specify the name of the script which is responsible for</td>
</tr>
<tr>
<td></td>
<td>translating IMSI to Global Title Address (GTA).</td>
</tr>
<tr>
<td>Timeout</td>
<td>Specifies the time (in seconds) to wait before an authentication request</td>
</tr>
<tr>
<td></td>
<td>times out; defaults to 15.</td>
</tr>
<tr>
<td>MaxTimeOuts</td>
<td>Maximum number of timeouts allowed for the remote server.</td>
</tr>
<tr>
<td>MaxSessionLimit</td>
<td>Maximum number of sessions allowed for the remote server.</td>
</tr>
<tr>
<td>ReactivateTimerInterval</td>
<td>Specifies the time interval (in milliseconds) to activate an inactive server;</td>
</tr>
<tr>
<td></td>
<td>defaults to 300000 ms (which is 5 minutes).</td>
</tr>
<tr>
<td>LimitOutstandingRequests</td>
<td>Required; the default is FALSE. Prime Access Registrar uses this property</td>
</tr>
<tr>
<td></td>
<td>in conjunction with the MaxOutstandingRequests property to tune the</td>
</tr>
<tr>
<td></td>
<td>RADIUS server’s use of the HLR.</td>
</tr>
<tr>
<td></td>
<td>When you set this property to TRUE, the number of outstanding requests for</td>
</tr>
<tr>
<td></td>
<td>this RemoteServer is limited to the value you specified in MaxOutstanding</td>
</tr>
<tr>
<td></td>
<td>Requests. When the number of requests exceeds this number, Prime Access</td>
</tr>
<tr>
<td></td>
<td>Registrar queues the remaining requests, and sends them as soon as the</td>
</tr>
<tr>
<td></td>
<td>number of outstanding requests drops to this number.</td>
</tr>
<tr>
<td>MaxOutstandingRequests</td>
<td>Required when you have set the LimitOutstandingRequests to TRUE. The number</td>
</tr>
<tr>
<td></td>
<td>you specify, which must be greater than zero, determines the maximum</td>
</tr>
<tr>
<td></td>
<td>number of outstanding requests allowed for this remote server.</td>
</tr>
<tr>
<td>TrafficMode</td>
<td>The mode of the traffic for the HLR. The possible values are LOADSHARE or</td>
</tr>
<tr>
<td></td>
<td>ACTSTANDBY.</td>
</tr>
<tr>
<td>LoadShareMode</td>
<td>Required. The TrafficMode is set as LOADSHARE, which is a type of load</td>
</tr>
<tr>
<td></td>
<td>sharing scheme.</td>
</tr>
<tr>
<td></td>
<td>When there is more than one associations with HLR, then the load sharing</td>
</tr>
<tr>
<td></td>
<td>is set as Signaling Link Selection (SLS). SLS is done based on a simple</td>
</tr>
<tr>
<td></td>
<td>round-robin basis.</td>
</tr>
<tr>
<td>MAPVersion</td>
<td>The version of the MAP. The possible values are 2 or 3. Specify the MAP</td>
</tr>
<tr>
<td></td>
<td>version that the HLR supports, i.e, 2 or 3 during the configuration.</td>
</tr>
<tr>
<td>NetworkVariant</td>
<td>Required. Choose ITU or ANSI to represent the network variant switch.</td>
</tr>
<tr>
<td>SubServiceField</td>
<td>Specifies the type of network to which this SAP belongs. The possible</td>
</tr>
<tr>
<td></td>
<td>options are:</td>
</tr>
<tr>
<td></td>
<td>• INT—represents international network</td>
</tr>
<tr>
<td></td>
<td>• NAT—represents national network</td>
</tr>
<tr>
<td></td>
<td>• RESERVE—represents reserved network</td>
</tr>
<tr>
<td></td>
<td>• SPARE—represents spare network</td>
</tr>
</tbody>
</table>
### Table 23-1  
**SIGTRAN-M3UA Stack Properties (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| SCCPVariant            | The Signaling Connection Control Part (SCCP) variant of the Global Title:  
                          - Select **ITU88**, **ITU92**, or **ITU96**, if **NetworkVariant** is set to ITU.  
                          - Select **ANS88**, **ANS92**, or **ANS96**, if **NetworkVariant** is set to ANS.  |
| TCAPVariant            | Required; represents the name of the tcap network variant switch. The possible options are ITU88, ITU92, or ITU96.                               |
| NetworkAppearance      | Required. A parameter that represents network appearance in the M3UA packet. Value ranges from 0-2147483647 and the default value is 1.  
                          This is optional as per the RFC 4666 (http://tools.ietf.org/html/rfc4666.)  
                          You can set this value to 0 to remove network appearance from the data packet.    |
| NetworkIndicator       | The network indicator used in SCCP address. The possible options are NAT and INT which represents international network and national network respectively. |
| MLCNumber              | Required, if you select FetchAuthorizationInfo as True in EAP-AKA or EAP-SIM services. Also, required for M3UA service for fetching the MSISDN from the HLR.  
                          This is the map layer network node number by which the HLR identifies the Prime Access Registrar in the network. The MLC number is configured in E.164 format.  
                          **Note** MLC is a max-15 digit number.                                                                 |
| RoutingIndicator       | Required; represents the routing indicator. The possible values are Route on Global Title (RTE_GT) or Route on Sub System Number (RTE_SSN).  
                          You can use either RTE_GT or RTE_SSN value to route the packets for HLR.          |
| RoutingParameters      |                                                                                                                                              |
| OriginPointCode        | Required; represents the originating point of a message in a signaling network. The value ranges from 0-16777215.                                  
                          This value must be less than OPCMask.                                                                 |
| DestinationPointCode   | Required; represents the destination address of a signaling point in a SS7 network.                                                          
                          This value must be less than DPCMask.                                                                 |
| RemoteSubSystemNumber  | Required; represents the sub system number of the remote server. The RemoteSubSystemNumber is set as 6 by default.                             |
| OPCMask                | Represents the wild card mask for the origin point code. The value ranges from 0-16777215.  
                          Default value is 16383 for ITU and16777215 for ANSI.                                     |
| DPCMask                | Represents the wild card mask for the destination point code. The value ranges from 0-16777215.  
                          Default value is 16383 for ITU and16777215 for ANSI.                                     |
| ServiceIndicatorOctet  | Represents the service identifier octet. The value ranges from 0-255.                                                                         |
Table 23-1 SIGTRAN-M3UA Stack Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RoutingContext</td>
<td>Required; represents the routing context which ranges from 0-16777215.</td>
</tr>
<tr>
<td>SourceIPAddresses</td>
<td>Represent the multiple source IP addresses configured on the remote server.</td>
</tr>
<tr>
<td>DestinationIPAddresses</td>
<td>Represent the multiple destination IP addresses configured on the remote server.</td>
</tr>
<tr>
<td>SourceGTAddress</td>
<td>The following fields are displayed only when you set RTE_GT as RoutingIndicator.</td>
</tr>
<tr>
<td>SourceGTDigits</td>
<td>Required; an unique number to identify the source.</td>
</tr>
<tr>
<td>SourceGTFormat</td>
<td>Required; represents the format of the global translation (GT) rule. The possible values are GTFRMT_0, GTFRMT_1, GTFRMT_2, GTFRMT_3, GTFRMT_4, or GTFRMT_5. The GT format is GTFRMT_0, GTFRMT_1, or GTFRMT_2 for ANSI variant. GTFRMT_0 is the default format for both ANSI and ITU variants.</td>
</tr>
<tr>
<td>SourceNatureofAddress</td>
<td>Required; represents the type of the source address. The possible values are ADDR_NOTPRSNT (Address not present), SUBNUM (Subscriber number), NATSIGNUM (National significant number), or INTNUM (International number.)</td>
</tr>
<tr>
<td>SourceTranslationType</td>
<td>Required; represents the type of translation. The possible values ranges from 0-255.</td>
</tr>
<tr>
<td>SourceNumberingPlan</td>
<td>Required; represents the numbering plan of the network that the subscriber uses. For example, land mobile numbering plan, ISDN mobile numbering plan, private or network specific numbering plan.</td>
</tr>
<tr>
<td>SourceEncodingScheme</td>
<td>Required; represents the BCD encoding scheme. The possible values are UNKN (Unknown), BCDODD (BCD Odd), BCDEVEN (BCD Even), or NWSPEC (National specific). This must be set based on the length of the GT.</td>
</tr>
<tr>
<td>DestinationGTAddress</td>
<td>The following fields are displayed only when you set RTE_GT as RoutingIndicator.</td>
</tr>
<tr>
<td>DestGTDigits</td>
<td>Required; an unique number to identify the destination.</td>
</tr>
<tr>
<td>DestGTFormat</td>
<td>Required; represents the format of the global translation (GT) rule. The possible values are GTFRMT_0, GTFRMT_1, GTFRMT_2, GTFRMT_3, GTFRMT_4, or GTFRMT_5. The GT format is GTFRMT_0, GTFRMT_1, or GTFRMT_2 for ANSI variant. GTFRMT_0 is the default format for both ANSI and ITU variants.</td>
</tr>
<tr>
<td>DestNatureofAddress</td>
<td>Required; represents the type of the destination address. The possible values are ADDR_NOTPRSNT (Address not present), SUBNUM (Subscriber number), NATSIGNUM (National significant number), or INTNUM (International number.)</td>
</tr>
<tr>
<td>DestTranslationType</td>
<td>Required; represents the type of translation. The possible values ranges from 0-255.</td>
</tr>
</tbody>
</table>
Configuring EAP-AKA/EAP-SIM with SIGTRAN-M3UA

The following shows an example configuration of SIGTRAN-M3UA remote server with ANSI variant:

```
[ //localhost/Radius/RemoteServers ]
Entries 1 to 1 from 1 total entries
Current filter: <all>

STP/
Name =STP
Description =
Protocol = Sigtran-m3ua
SourcePort = 2905
LocalSubSystemNumber = 149
DestinationPort = 2905
IMSITranslationScript~ =
Timeout = 15
MaxTimeOuts = 200
MaxSessionLimit = 0
ReactivateTimerInterval = 2000
LimitOutstandingRequests = FALSE
MaxOutstandingRequests = 0
MAPVersion = 2
NetworkVariant = ANS
SubServiceField = NAT
SCCPVariant = ANS92
TCAPVariant = ITU96
NetworkAppearance = 1
NetworkIndicator = NAT
MLCNumber = 123456789012345
TrafficMode = LOADSHARE
LoadShareMode = SLS
RoutingIndicator = RTE_GT
GlobalTitleTranslationScript~ =
MaskPointCode = FALSE
RoutingParameters/
  OriginPointCode = 13967019
  DestinationPointCode = 13966849
  RemoteSubSystemNumber = 6
  OPCMask = 167777215
  DPCMask = 167777215
  ServiceIndicatorOctet = 3
  RoutingContext = 11
SourceIPAddresses/
  1. 10.81.78.142
DestinationIPAddresses/
  1. 10.81.78.145
SourceGTAddress/
  SourceGTDigits = 919845071842
```

Table 23-1 SIGTRAN-M3UA Stack Properties (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DestNumberingPlan</td>
<td>Required; represents the numbering plan of the network that the subscriber uses. For example, Land mobile numbering plan, ISDN mobile numbering plan, private or network specific numbering plan. Possible values are DATA, GENERIC, ISDN, ISDNMOB, LANMOB, MARMOB, NWSPEC, TEL, TELEX, and UNKN.</td>
</tr>
<tr>
<td>DestEncodingScheme</td>
<td>Required; represents the BCD encoding scheme. The possible values are UNKN (Unknown), BCDODD (BCD Odd), BCDEVEN (BCD Even), or NWSPEC (National specific). This must be set based on the length of the GT.</td>
</tr>
</tbody>
</table>

The DestNumberingPlan property represents the numbering plan of the network that the subscriber uses. Possible values are DATA, GENERIC, ISDN, ISDNMOB, LANMOB, MARMOB, NWSPEC, TEL, TELEX, and UNKN. The DestEncodingScheme property represents the BCD encoding scheme. Possible values are UNKN (Unknown), BCDODD (BCD Odd), BCDEVEN (BCD Even), or NWSPEC (National specific). This must be set based on the length of the GT.
Blacklisting IMSI Values

Prime Access Registrar allows you to blacklist one or more IMSI values available in the EAP-SIM or EAP-AKA requests forwarded to an HLR. A scripting point option is provided such that you can set an environment dictionary variable Blacklisted-IMSI to **TRUE** or **FALSE** to blacklist or whitelist IMSI values respectively. An IMSI value marked as blacklisted is rejected and will not be forwarded to the HLR.

To blacklist an IMSI value:

**Step 1** Configure a SIGTRAN-M3UA remote server as described in Configuring SIGTRAN-M3UA Remote Server, page 23-6.

**Step 2** Configure an EAP-AKA or EAP-SIM service with the SIGTRAN-M3UA remote server. See Configuring EAP-AKA/EAP-SIM with SIGTRAN-M3UA, page 23-4 for an example.

**Step 3** Run the following script:

```tcl
proc CheckBlackList {request response environ} {
    set imsi [ $environ get IMSI ]
    if { [ string compare $imsi 984579621012345 ] == 0 } {
        $environ put Blacklisted-IMSI TRUE
        $environ put Notification-Code 19384
    }
}
```

Where, **CheckBlackList** is the entrypoint variable of the global title translation script **checklist**, as shown in the example below:

```tcl
[ //localhost/Radius/Scripts/checklist ]
Name = checklist
Description =
Language = tcl
Filename = tclscript.tcl
EntryPoint = CheckBlackList
InitEntryPoint =
InitEntryPointArgs =
```

If the environment variable **Blacklisted-IMSI** is set as **TRUE** and if the IMSI value available in the incoming script matches the given string, then that IMSI is blacklisted and will not be forwarded to the HLR. You can configure a notification code to represent failure. If no notification code is set, 16384 representing **General Failure** is sent upon rejection of an IMSI value. For more information about the Notification-Code variable, see Appendix B, “Environment Dictionary.”

Additionally, you can configure the script to compare the incoming IMSI against the configured IMSIs and take appropriate action as to whether blacklist or whitelist the incoming IMSI.
Configuring M3UA Service

Prime Access Registrar supports the M3UA service, which is used to fetch MSISDN from IMSI or vice versa through RADIUS packets, see Chapter 4, “M3UA,” for more information.

To configure the M3UA service with SIGTRAN-M3UA remote server:

**Step 1** Launch `aregcmd`.

**Step 2** Create an M3UA service.

```
cd /Radius/Services
add FetchAuthInfo
```

**Step 3** Set the type as M3UA.

```
set type M3UA
```

**Step 4** Set `AuthorizationInfoLookUp` to one of the following:

- MSISDN-IMSI—To fetch MSISDN in the request and send IMSI in the response to the HLR.
- IMSI-MSISDN—To fetch IMSI in the request and send MSISDN in the response to the HLR.

![Note](Image)

See Example Configuration, page 23-14 for a sample configuration with

- Map-Restore—To fetch the profile information of a subscriber from the HLR. For more information on configuring the M3UA service with Map Restore Data authorization, see Configuring M3UA Service with Map Restore Data Authorization, page 23-15.

```
set AuthorizationInfoLookUp IMSI-MSISDN
```

**Step 5** Add M3UA remote server in the remoteServers.

```
cd remoteServers
Set 1 m3ua
```

### Example Configuration

The following shows an example configuration of the M3UA service:

```bash
[ //localhost/Radius/Services/test ]
Name = test
Description =
Type = m3ua
IncomingScript~ =
OutgoingScript~ =
OutageScript~ =
OutagePolicy~ = RejectAll
AuthorizationInfoLookUp = IMSI-MSISDN
RemoteServers/
```
Configuring M3UA Service with Map Restore Data Authorization

Prime Access Registrar provides the Map Restore Data functionality to fetch the profile information of a subscriber from the HLR.

This topic contains the following sections:
- Map Restore Data Authorization Flow, page 23-15
- CS Insert Subscriber Data Structure, page 23-16
- CLI Configuration for Map-Restore-Data, page 23-17

Map Restore Data Authorization Flow

Prime Access Registrar sends a MAP_SEND_AUTH_INFO request to HLR on receiving EAP-SIM / EAP-AKA authentication request and fetches the authentication vectors in MAP_SEND_AUTH_INFO_RES message. Prime Access Registrar checks the IMSI and if it is authentic, sends a MAP_RESTORE_DATA_REQUEST to fetch the profile information from the HLR. HLR then responds with MAP_INSERT_SUBSCRIBER_DATA request to Prime Access Registrar. The request contains the circuit switched (CS) profile information for a subscriber.

Prime Access Registrar server stores the profile information based on the ProfileInfo configuration and sends a MAP_INSERT_SUBSCRIBER_DATA_RESPONSE to HLR. HLR responds with MAP_RESTORE_DATA_RESPONSE to Prime Access Registrar. After successful acknowledgment of MAP_RESTORE_DATA, Prime Access Registrar server maps the fetched profile through RestoreDataMappings to any of the environment variables configured by the user. The CS profile used to authorize WI-FI access which is fetched from HLR can be transported to access point in any of the radius attribute.

The mapping of the values in the response to a profile is possible based on the configuration in the profilemappings configuration.

Figure 23-2 represents the Map-Restore-Data message flow between Prime Access Registrar and HLR.
Configuring M3UA Service

Figure 23-2 Map-Restore-Data Authorization Flow

CS Insert Subscriber Data Structure

Figure 23-3 shows the parameters fetched by Prime Access Registrar on receipt of the subscriber data request.

Figure 23-3 CS Insert Subscriber Data Structure
CLI Configuration for Map-Restore-Data

If you set AuthorizationInfoLookUp to Map-Restore, two additional properties ProfileMappings and RestoreDataMappings are displayed.

The restore data mapping parameters include LSA information, LCS information, and subscriber data. You can configure an index with a value or a range to fetch one or more properties from the subscriber data.

The following is an example configuration of an M3UA service with Map-Restore-Data authorization:

```plaintext
[ //localhost/Radius/Services/serv1 ]
Name = serv1
Description =
Type = m3ua
IncomingScript- =
OutgoingScript- =
OutageScript- =
OutagePolicy- = RejectAll
AuthorizationInfoLookUp = MAP-RESTORE
RemoteServers/
  1. server1
RestoreDataMappings/
  IMSI = imsi
  Naea-PreferredCI = naea
  RoamingRestrictedInSgsnDueToUnsupportedFeature =
  NetworkAccessMode =
  LMUIndicator =
  ISTAlertTimer =
  SuperChargerSupportedInHLR =
  CSAllocationRetentionPriority =
  ChargingCharacteristics =
  AccessRestrictionData =
  UE-ReachabilityRequestIndicator =
  Category =
LSAInformation/
  CompletedDataListIncluded = completeddatalist
  LSAOnlyAccessIndicator =
  LSADataList/
    Index = 6
    LSAIdentity = lsaid
    LSAAttributes = lsattrib
    LSAActiveModeIndicator = activmode
SubscriberData/
  MSISDN = msisdn
  SubscriberStatus = substatus
  RoamingRestrictionDueToUnsupportedFeature =
  BearerServiceList/
    Index = 6-10
    BearerService = bearsrvc
  TeleServiceList/
    Index =
    TeleService =
  ProvisionedSS/
    Index = 4-6
  ForwardingInfo/
    FI-SS-Code = fisscode
  ForFeatureList/
    Index = 7-10
    FF-SS-Status = ffssstatus
    ForwardedToNumber =
    ForwardedToSubaddress =
    ForwardingOptions =
```
NoReplyConditionTime =
LongForwardedToNumber =
BasicService/
  BS-Ext-BearerService = bextbsservice
  BS-Ext-Teleservice = bextteleservice
CallBarringInfo/
  CB-SS-Code =
CallBarFeatureList/
  Index =
  CB-SS-Status =
BasicService/
  CB-Ext-BearerService =
  CB-Ext-Teleservice =
CugInfo/
  CugSubList/
    Index =
    CugSubscription/
      Cug-Index =
      cug-Interlock =
      IntraCUG-Options =
      BasicServiceGroupList/
        Index =
        CUG-Ext-BearerService =
        CUG-Ext-Teleservice =
CugInformation/
  Cug-FeatureList/
    Index =
    CUG-Feature/
      BasicService.Ext-BearerService =
      PreferentialCUG-Indicator =
      InterCUG-Restrictions =
SS-Data/
  SSD-SS-Code =
  SSD-SS-Status =
SS-SubscriptionOption/
  CliRestrictionOption =
  OverrideCategory =
BasicServiceGroupList/
  Index =
  BSG-Ext-BearerService =
  BSG-Ext-Teleservice =
EMLPP-Info/
  MaximumEntitledPriority =
  DefaultPriority =
ODB-Data/
  ODB-GeneralData =
  ODB-HPLMN-Data =
RegionalSubscriptionData/
  Index =
  RegionalSubscriptionData =
VBSSubscriptionData/
  Index =
  VBS-GroupId =
  BroadcastInitEntitlement =
VGCSSubscriptionData/
  Index =
  VGCS-GroupId =
  AdditionalSubscriptions =
  AdditionalInfo =
  LongGroupId =
LCSInformation/
  GMLC-List/
    Index =
    GMLC =
LCS-PrivacyExceptionList/
  Index =
  PE-SS-Code =
  SS-Status =
  LCSNotificationToMSUser =
ExternalClientList/
  Index =
  ClientIdentity.ExternalAddress =
  ExtCliGMLC-Restriction =
  ExtCliNotificationToMSUser =
PLMNClientList/
  Index =
  PLMNClient =
ServiceTypeList/
  Index =
  ServiceTypeIdentity =
  ServiceTypeGMLC-Restriction =
  ServiceTypeNotificationToMSUser =
MOLR-List/
  Index =
  MOLR-SS-Code =
  MOLR-SS-Status =
MC-SS-Info/
  MC-SS-Code =
  MC-SS-Status =
  NbrSB =
  NbrUser =
SGSN-CAMEL-SubscriptionInfo/
GPRS-CI/
  GPRS-CamelCapabilityHandling =
  GPRS-NotificationToCSE =
  GPRS-CI-Active =
  GPRS-CamelTDPDataList/
    Index =
    GPRS-TriggerDetectionPoint =
    GPRS-ServiceKey =
    GPRS-GSMSCF-Address =
    DefaultSessionHandling =
MO-SMS-CI/
  MOSMS-CamelCapabilityHandling =
  MOSMS-NotificationToCSE =
  MOSMS-CI-Active =
  SMS-CAMEL-TDP-DataList/
    Index =
    MO-SMS-TriggerDetectionPoint =
    MO-ServiceKey =
    MO-GSMSCF-Address =
    MO-DefaultSMSHandling =
MT-SMS-CI/
  MTSMS-CamelCapabilityHandling =
  MTSMS-NotificationToCSE =
  MTSMS-CI-Active =
  SMS-CAMEL-TDP-DataList/
    Index =
    MT-SMS-TriggerDetectionPoint =
    MT-ServiceKey =
    MT-GSMSCF-Address =
    MT-DefaultSMSHandling =
MT-SMSCAMELTDP-CriteriaList/
  Index =
  SMS-TriggerDetectionPoint =
  TPDU-TypeCriterion =
MG-CI/
  MobilityTriggers =
Chapter 23      SIGTRAN-M3UA

Configuring M3UA Service

    MG-ServiceKey =
    MG-GSMSCF-Address =
    MG-NotificationToCSE =
    MG-CSI-Active =

ProfileMappings/
   imsi = 100,Profile1
   naea = 20,Profile2
   naea = 30,Profile3

[ //localhost/RADIUS/Profiles ]
    Entries 1 to 6 from 6 total entries
    Current filter: <all>

   default-PPP-users/
   default-SLIP-users/
   default-Telnet-users/
   Profile1/
   Profile2/
   Profile3/

Table 23-2 shows the restore data mapping parameters.

Table 23-2  Restore Data Mappings and Profile Mappings Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMSI</td>
<td>IMSI received in the response from HLR.</td>
</tr>
<tr>
<td>Naea-Preferred CI</td>
<td>North American Equal Access preferred Carrier ID List. A list of the preferred carrier identity codes that are subscribed to.</td>
</tr>
<tr>
<td>Roaming Restricted In Sgsn Due To Unsupported Feature</td>
<td>Indicates that a subscriber is not allowed to roam in the current Service GPRS Support Node (SGSN) or Cisco Mobility Management Entity (MME) area.</td>
</tr>
<tr>
<td>Network Access Mode</td>
<td>The Network Access Mode (NAM) defines if the subscriber is registered to get access to the CS (non-GPRS/EPS network), to the PS (GPRS/EPS) network or to both networks. NAM describes the first level of the subscriber data pseudo-tree below the IMSIroot. It is permanent subscriber data stored in the HSS / HLR and the SGSN with the Gs interface option, and the MME with the SGs interface option.</td>
</tr>
<tr>
<td>LMU Indicator</td>
<td>Indicates the presence of an LMU.</td>
</tr>
<tr>
<td>IST Alert Timer</td>
<td>Indicates the IST alert timer value that must be used in the Mobile Switching Center (MSC) to inform the HLR about the call activities that the subscriber performs.</td>
</tr>
<tr>
<td>Super Charger Supported In HLR</td>
<td>Indicates whether super charger concept is supported in HLR.</td>
</tr>
<tr>
<td>CS Allocation Retention Priority</td>
<td>Allocation-retention priority for Circuit Switched (CS). This parameter specifies relative importance to compare with other bearers about allocation and retention of bearer.</td>
</tr>
<tr>
<td>Charging Characteristics</td>
<td>Subscribed charging characteristics.</td>
</tr>
</tbody>
</table>
### Chapter 23      SIGTRAN-M3UA

#### Configuring M3UA Service

**Table 23-2  Restore Data Mappings and Profile Mappings Parameters (continued)**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Restriction Data</td>
<td>Allowed Recipient Access Table (RAT) according to subscription data.</td>
</tr>
<tr>
<td>UE Reachability Request Indicator</td>
<td>Indicates that the Home Subscriber Server (HSS) is awaiting a notification of user equipment (UE) reachability.</td>
</tr>
<tr>
<td>Category</td>
<td>Calling party category</td>
</tr>
<tr>
<td>LSA Information</td>
<td>These parameters refer to one or more localized service areas (LSAs) a subscriber may be a member of, together with the priority, the preferential access indicator, the active mode support indicator and active mode indication of each localized service area. The access right outside these localized service areas is also indicated.</td>
</tr>
</tbody>
</table>

#### Subscriber Data

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSISDN</td>
<td>MSISDN value in the subscriber data.</td>
</tr>
<tr>
<td>Subscriber Status</td>
<td>Barring status of the subscriber, which could be Service Granted or Operator Determined Barring.</td>
</tr>
<tr>
<td>Roaming Restriction Due To Unsupported Feature</td>
<td>Indicates that the subscriber is not allowed to roam in the current MSC area.</td>
</tr>
<tr>
<td>Bearer Service List</td>
<td>List of extensible bearer services subscribed. Configure the index value to fetch only the required bearer services.</td>
</tr>
<tr>
<td>TeleService List</td>
<td>List of extensible teleservices subscribed. Configure the index value to fetch only the required teleservices.</td>
</tr>
<tr>
<td>Provisioned SS</td>
<td>List of supplementary services provisioned. Configure the index value to fetch only the required supplementary services.</td>
</tr>
<tr>
<td>ODB-Data</td>
<td>Operator Determined Barring (ODB) general data and ODB Home Public Land Mobile Network (HPLMN) specific data.</td>
</tr>
<tr>
<td>Regional Subscription Data</td>
<td>List of regional subscription areas (zones) in which the subscriber is allowed to roam. Configure the index value to fetch only the required zones.</td>
</tr>
<tr>
<td>VBS Subscription Data</td>
<td>List of Voice Broadcast Services (VBS) subscribed. Configure the index value to fetch only the required VBS.</td>
</tr>
<tr>
<td>VGCS Subscription Data</td>
<td>List of Voice Group Call Services (VGCS) subscribed. Configure the index value to fetch only the required VGCS.</td>
</tr>
</tbody>
</table>

#### LCS Information

Live Communication Server (LCS) related information for the subscriber.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMLC-List</td>
<td>List of Gateway Mobile Location Centers (GMLCs) that are permitted to issue a call/session unrelated or call/session related MT-LR request. Configure the index value to fetch only the required GMLCs.</td>
</tr>
<tr>
<td>LCS-Privacy Exception List</td>
<td>Classes of LCS client that are allowed to locate any target Mobile Station (MS). Configure the index value to fetch only the required classes.</td>
</tr>
</tbody>
</table>
Support for SCTP Multihoming in SIGTRAN-M3UA

Stream Control Transmission Protocol (SCTP) is an IP transport protocol that supports data exchange between exactly two endpoints. Multihoming feature of SCTP provides the ability for a single SCTP endpoint to support multiple IP addresses. With this feature, each of the two endpoints during an SCTP association can specify multiple points of attachment. Each endpoint will be able to receive messages from any of the addresses associated with the other endpoint. With the use of multiple interfaces, data can be sent to alternate addresses when failures occur and thus Prime Access Registrar runs successfully even during network failures.

Prime Access Registrar allows you to configure multiple source and destination addresses on the remote server. The following shows an example configuration of SIGTRAN-M3UA remote server with multiple source and destination addresses:

```
[ /Radius/RemoteServers/m3ua ]
  Name = m3ua
  Description = Protocol = sigtran-m3ua
  SourcePort = 2805
  LocalSubSystemNumber = 149
  DestinationPort = 2855
  IMSITranslationScript~ =
  GlobalTitleTranslationScript~ =
  Timeout = 15
  ReactivateTimerInterval = 300000
  LimitOutstandingRequests = FALSE
```

### Configuring Environment Variables to Fetch Subscriber Data Values

You can configure an environment variable to fetch the required values from the subscriber data packets. You can run a script to fetch the environment variable along with the values. See the example below:

```bash
proc FetchBearerService {request response environ} {
  set bearerService [ $environ get bs-ext ]
  $request trace 2 "BearerService value fetched is " $bearerService
}
```

In the above script bs-ext is the environment variable that is configured. If the values fetched from BearerServiceList are 17,18,19,20 and 21, the above script returns the value 17:18:19:20:21.

Similarly we can run scripts to retrieve other environment variables as well.

### Support for SCTP Multihoming in SIGTRAN-M3UA

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOLR-List</td>
<td>Code and status of Mobile Originating Location Request (MO-LR) subscribed.</td>
</tr>
<tr>
<td></td>
<td>Configure the index value to fetch only the required requests.</td>
</tr>
<tr>
<td>MC-SS-Info</td>
<td>Parameters identifying Multicall (MC) supplementary services (SS).</td>
</tr>
<tr>
<td>SGSN-CAMEL-Subscription Info</td>
<td>Parameters identifying the subscribers as having Customized Application for Mobile Enhanced Logic (CAMEL) services that are invoked in the SGSN.</td>
</tr>
<tr>
<td>ProfileMappings</td>
<td>Attribute: The RADIUS attribute to map the fetched profile data.</td>
</tr>
<tr>
<td></td>
<td>Value:Profile Value of the attribute.</td>
</tr>
</tbody>
</table>

**Table 23-2  Restore Data Mappings and Profile Mappings Parameters (continued)**
MaxOutstandingRequests = 0
MAPVersion = 3
NetworkVariant = ITU
SubServiceField = NAT
TCAPVariant = ITU96
NetworkAppearance = 1
NetworkIndicator = NAT
MLCNumber = 123456789012345
TrafficMode = LOADSHARE
LoadShareMode = SLS
RoutingIndicator = RTE_SSNS
RoutingParameters/
OriginPointCode = 2
DestinationPointCode = 4
RemoteSubSystemNumber = 6
OPCMask = 16383
DPCMask = 16383
ServiceIndicatorOctet = 0
RoutingContext = 11
SourceIPAddresses/
DestinationIPAddresses/
--- cd SourceIPAddresses
--- add 1 192.168.0.2
--- add 2 192.168.0.3
--- cd ../DestinationIPAddresses
--- add 1 192.168.0.5
--- add 2 192.168.0.6

In the above example, the link between IP addresses 192.168.0.2 and 192.168.0.5 acts as the primary link and the link between IP addresses 192.168.0.3 and 192.168.0.6 acts as the secondary link. With the Multihoming feature, if one of the interfaces in the primary link is down, the secondary link carries the active traffic. On restoration of the IP address, the traffic switches back to the primary link.

Tuning Global SIGTRAN Parameters

Prime Access Registrar provides a CLI tool SigtranXMLEdit that allows you to edit the values of the global SIGTRAN XML parameters. The tool is available under the <installation directory>/bin directory, e.g. /cisco-ar/bin and the parameters are available in the default.xml file under the /cisco-ar/m3ua-cfg directory.

Table 23-3 lists the global SIGTRAN parameters that you can edit using the CLI tool.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rtoMin</td>
<td>Minimum value of retransmission timeout</td>
</tr>
<tr>
<td>rtoMax</td>
<td>Maximum value of retransmission timeout</td>
</tr>
<tr>
<td>rtoInitial</td>
<td>Initial value of retransmission timeout</td>
</tr>
<tr>
<td>alpha</td>
<td>Retransmission timeout alpha value</td>
</tr>
<tr>
<td>beta</td>
<td>Retransmission timeout beta value</td>
</tr>
<tr>
<td>maxAssocReTx</td>
<td>Maximum association retransmission</td>
</tr>
<tr>
<td>maxPathReTx</td>
<td>Maximum path retransmission</td>
</tr>
<tr>
<td>maxInitReTx</td>
<td>Maximum initial retransmission</td>
</tr>
</tbody>
</table>
To edit the SIGTRAN parameters:

**Step 1**
Launch the CLI tool **SIGTRANXMLEdit** from the `/cisco-ar/bin` directory.

The tool displays the list of editable parameters available in the `default.xml` file as shown below.

<table>
<thead>
<tr>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cookie life</td>
</tr>
<tr>
<td>Heartbeat interval</td>
</tr>
<tr>
<td>SACK period</td>
</tr>
<tr>
<td>Maximum number of inbound streams</td>
</tr>
<tr>
<td>Maximum number of outbound streams</td>
</tr>
<tr>
<td>Initial value of maximum transmission unit</td>
</tr>
<tr>
<td>Minimum Initial value of maximum transmission unit</td>
</tr>
<tr>
<td>Maximum Initial value of maximum transmission unit</td>
</tr>
</tbody>
</table>

The tool prompts you to enter the new value against the first parameter as shown below.

Enter values for the following parameters (just press ‘return’ to skip):

<table>
<thead>
<tr>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of inbound streams</td>
</tr>
</tbody>
</table>

In this example, 1024 is the value that exists for the parameter in the `default.xml` file.

**Step 2**
Type the new value and press ENTER or just press ENTER to skip and proceed to the next parameter. Perform this step for all parameters as shown below.

<table>
<thead>
<tr>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of inbound streams</td>
</tr>
<tr>
<td>Maximum number of outbound streams</td>
</tr>
<tr>
<td>Initial value of Maximum Transmission Unit</td>
</tr>
<tr>
<td>Minimum Initial value of Maximum Transmission Unit</td>
</tr>
<tr>
<td>Maximum initial Retransmission</td>
</tr>
<tr>
<td>Maximum association Retransmission</td>
</tr>
<tr>
<td>Maximum path Retransmission</td>
</tr>
<tr>
<td>Alpha value</td>
</tr>
<tr>
<td>Beta value</td>
</tr>
<tr>
<td>Initial value of Retransmission timeout</td>
</tr>
<tr>
<td>Minimum value of Retransmission timeout</td>
</tr>
<tr>
<td>Maximum value of Retransmission timeout</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of inbound streams</td>
</tr>
<tr>
<td>Maximum number of outbound streams</td>
</tr>
<tr>
<td>Initial value of Maximum Transmission Unit</td>
</tr>
<tr>
<td>Minimum Initial value of Maximum Transmission Unit</td>
</tr>
<tr>
<td>Maximum initial Retransmission</td>
</tr>
<tr>
<td>Maximum association Retransmission</td>
</tr>
<tr>
<td>Maximum path Retransmission</td>
</tr>
<tr>
<td>Alpha value</td>
</tr>
<tr>
<td>Beta value</td>
</tr>
<tr>
<td>Initial value of Retransmission timeout</td>
</tr>
<tr>
<td>Minimum value of Retransmission timeout</td>
</tr>
<tr>
<td>Maximum value of Retransmission timeout</td>
</tr>
</tbody>
</table>

---

**Table 23-3**
**Global SIGTRAN Parameter (continued)**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cookieLife</td>
<td>Cookie life</td>
</tr>
<tr>
<td>intervalTm</td>
<td>Heartbeat interval</td>
</tr>
<tr>
<td>maxAckDelayTm</td>
<td>SACK period</td>
</tr>
<tr>
<td>maxNmbInStrms</td>
<td>Maximum number of inbound streams</td>
</tr>
<tr>
<td>maxNmbOutStrms</td>
<td>Maximum number of outbound streams</td>
</tr>
<tr>
<td>mtuInitial</td>
<td>Initial value of maximum transmission unit</td>
</tr>
<tr>
<td>mtuMinInitial</td>
<td>Minimum Initial value of maximum transmission unit</td>
</tr>
<tr>
<td>mtuMaxInitial</td>
<td>Maximum Initial value of maximum transmission unit</td>
</tr>
</tbody>
</table>

---

Cisco Prime Access Registrar 7.0.1 User Guide
When prompted for a confirmation, type **Yes** and press ENTER to save the changes. The tool displays the modified parameters with the new and old values.

Changed Value of maxNmbInStrms is 87 <- 1024
Changed Value of maxNmbOutStrms is 90 <- 1024
Changed Value of mtuMinInitial is 65 <- 1500
Changed Value of mtuMaxInitial is 33 <- 1500
Changed Value of maxInitReTx is 9 <- 5
Changed Value of maxAssocReTx is 4 <- 10
Changed Value of maxPathReTx is 2 <- 5
Changed Value of alpha is 15 <- 12
Changed Value of beta is 34 <- 25
Changed Value of maxAckDelayTm is 89 <- 2
Changed Value of cookieLife is 67 <- 60
Changed Value of intervalTm is 89 <- 15

SIGTRAN-M3UA Logs

The following logs are applicable for SIGTRAN-M3UA:

- **stack.log**—Logs the interaction between Prime Access Registrar and STP/HLR.
- **sm.log**—Logs the internal debug information for SIGTRAN-M3UA stack manager.
- **m3ua.log**—Logs the inter-process communication between Prime Access Registrar and SIGTRAN-M3UA stack.
- **cliActivity.log**—Logs the initialization and command interactions.
Using SNMP

This chapter provides the following information about Cisco Prime Access Registrar (Prime Access Registrar) support for SNMP:

- Overview
- Supported MIBs
- SNMP Traps

Overview

Prime Access Registrar provides SNMP MIB and trap support for users of network management systems. The supported MIBs enable the network management station to collect state and statistic information from an Prime Access Registrar server. The traps enable Prime Access Registrar to notify interested network management stations of failure or impending failure conditions.

Prime Access Registrar supports the MIBs defined in the following RFCs:

- RADIUS Authentication Client MIB for IPv6, RFC 4668
- RADIUS Authentication Server MIB for IPv6, RFC 4669
- RADIUS Accounting Client MIB for IPv6, RFC 4670
- RADIUS Accounting Server MIB for IPv6, RFC 4671
- CISCO Diameter Base Protocol MIB

Prime Access Registrar MIB support enables a standard SNMP management station to check the current state of the server as well as the statistics on each client or each proxied remote server.

Prime Access Registrar Trap support enables a standard SNMP management station to receive trap messages from an Prime Access Registrar server. These messages contain information indicating that either the server was brought up or down, or that the proxied remote server is down or has come back online.

Supported MIBs

The MIBs supported by Prime Access Registrar enable a standard SNMP management station to check the current state of the server and statistics for each client or proxied remote server.
This section contains the following topics:

- **RADIUS-AUTH-CLIENT-MIB**
- **RADIUS-AUTH-SERVER-MIB**
- **RADIUS-ACC-CLIENT-MIB**
- **RADIUS-ACC-SERVER-MIB**
- **CISCO-DIAMETER-BASE-PROTOCOL-MIB**
- Diameter SNMP and Statistics Support
- TACACS+ SNMP and Statistics Support

**RADIUS-AUTH-CLIENT-MIB**

The RADIUS-AUTH-CLIENT-MIB describes the client side of the RADIUS authentication protocol. The information contained in this MIB is useful when an Prime Access Registrar server is used as a proxy server.

**RADIUS-AUTH-SERVER-MIB**

The RADIUS-AUTH-SERVER-MIB describes the server side of the RADIUS authentication protocol. The information contained in this MIB describes managed objects used for managing a RADIUS authentication server.

**RADIUS-ACC-CLIENT-MIB**

The RADIUS-ACC-CLIENT-MIB describes the client side of the RADIUS accounting protocol. The information contained in this MIB is useful when an Prime Access Registrar server is used for accounting.

**RADIUS-ACC-SERVER-MIB**

The RADIUS-ACC-CLIENT-MIB describes the server side of the RADIUS accounting protocol. The information contained in this MIB is useful when an Prime Access Registrar server is used for accounting.

**CISCO-DIAMETER-BASE-PROTOCOL-MIB**

Prime Access Registrar uses the CISCO-DIAMETER-BASE-PROTOCOL-MIB as an interface to query the Diameter statistics, though configuring the Diameter through SNMP is not possible. Prime Access Registrar supports LocalStatistics and PeerStatistics only. The LocalStats provides statistical information about the local diameter server and the PeerStats provides statistical information about the peers and the messages to/from the peers.
Diameter SNMP and Statistics Support

Prime Access Registrar also supports Diameter SNMP MIB (CISCO-DIAMETER-BASE-PROTOCOL-MIB) to describe the Diameter Base Protocol statistics.

Prime Access Registrar supports statistic of Diameter messages to include the additional counters. This is supported through the CLI/GUI and SNMP. The diameter statistics includes peer statistics and global summary statistics details.

TACACS+ SNMP and Statistics Support

Prime Access Registrar supports the CISCO-AAA-SERVER-MIB to describe the statistics of TACACS+ protocol. TACACS+ protocol is used to authenticate an user via various services such as login services see TACACS+ Support for AAA for more information. This is supported through the CLI/GUI and SNMP.

SNMP Traps

The traps supported by Prime Access Registrar enable a standard SNMP management station to receive trap messages from an Prime Access Registrar server. These messages contain information indicating whether a server was brought up or down, or that the proxied remote server is down or has come back online.

A trap is a network message of a specific format issued by an SNMP entity on behalf of a network management agent application. A trap is used to provide the management station with an asynchronous notification of an event.

When a trap is generated, a single copy of the trap is transmitted as a trap PDU to each destination contained within a list of trap recipients.

The list of trap recipients is shared by all events and is determined at server initialization time along with other trap configuration information. The list of trap recipients dictates where Prime Access Registrar traps are directed.

The configuration of any other SNMP agent on the host is ignored. By default, all traps are enabled but no trap recipients are defined. By default, no trap is sent until trap recipients are defined.

Traps are configured using the command line interface (CLI). After configuring traps, the configuration information is re initialized when a server reload or restart occurs.

Note

SNMP queries and traps communication can be performed over IPv6.

When you configure traps, you must provide the following information:

- List of trap recipients (community string for each)
- Suppressing traps for any type of message
- Frequency of traps for any type of message

This section contains the following topics:

- Supported Traps
Supported Traps

The traps supported by Prime Access Registrar enable the Prime Access Registrar server to notify interested management stations of events, failure, or impending failure conditions. Traps are a network message of a specific format issued by an SNMP entity on behalf of a network management agent application. Traps are used to provide the management station with an asynchronous notification of an event.

This section contains the following topics:

- carServerStart
- carServerStop
- carInputQueueFull
- carInputQueueNotVeryFull
- carOtherAuthServerNotResponding
- carOtherAuthServerResponding
- carOtherAccServerNotResponding
- carOtherAccServerResponding
- carAccountingLoggingFailure
- carLicenseUsage
- carSigtranLicenseUsage
- carDiameterPeerDown
- carDiameterPeerUp

**carServerStart**

carServerStart signifies that the server has started on the host from which this notification was sent. This trap has one object, `carNotifStartType`, which indicates the start type. A `firstStart` indicates this is the server process’ first start. `reload` indicates this server process has an internal reload. This typically occurs after rereading some configuration changes, but `reload` indicates this server process did not quit during the reload process.

**carServerStop**

carServerStop signifies that the server has stopped normally on the host from which this notification was sent.

**carInputQueueFull**

carInputQueueFull indicates that the percentage of use of the packet input queue has reached its high threshold. This trap has two objects:

- `carNotifInputQueueHighThreshold`—indicates the high limit percentage of input queue usage
Chapter 24 Using SNMP

SNMP Traps

- `carNotifInputQueueLowThreshold`—indicates the low limit percentage of input queue usage

By default, `carNotifInputQueueHighThreshold` is set to 90% and `carNotifInputQueueLowThreshold` is set to 60%.

**Note**
The values for these objects cannot be changed at this time. You will be able to modify them in a future release of Prime Access Registrar.

After this notification has been sent, another notification of this type will not be sent again until the percentage usage of the input queue goes below the low threshold.

If the percentage usage reaches 100%, successive requests might be dropped, and the server might stop responding to client requests until the queue drops down again.

### `carInputQueueNotVeryFull`

`carInputQueueNotVeryFull` indicates that the percentage usage of the packet input queue has dropped below the low threshold defined in `carNotifInputQueueLowThreshold`. This trap has two objects:

- `carNotifInputQueueHighThreshold`—indicates the high limit percentage of input queue usage
- `carNotifInputQueueLowThreshold`—indicates the low limit percentage of input queue usage

After this type of notification has been sent, it will not be sent again until the percentage usage goes back up above the high threshold defined in `carNotifInputQueueHighThreshold`.

### `carOtherAuthServerNotResponding`

`carOtherAuthServerNotResponding` indicates that an authentication server is not responding to a request sent from this server. This trap has three objects:

- `radiusAuthServerAddress`—indicates the identity of the concerned server
- `radiusAuthClientServerPortNumber`—indicates the port number of the concerned server
- `carAuthServerType`—indicates the type of the concerned server

The index of these three objects identifies the entry in `radiusAuthServerTable` and `carAccServerExtTable` which maintains the characteristics of the concerned server.

**Note**
One should not rely solely on `carOtherAuthServerNotResponding` for server state. Several conditions, including a restart of the Prime Access Registrar server, could result in either multiple `carOtherAuthServerNotResponding` notifications being sent or in a `carOtherAuthServerResponding` notification not being sent. NMS can query the `carAuthServerRunningState` in `carAuthServerExtTable` for the current running state of this server.

### `carOtherAuthServerResponding`

`carOtherAuthServerResponding` signifies that an authentication server which had formerly been in a down state is now responding to requests from the Prime Access Registrar server. This trap has three objects:

- `radiusAuthServerAddress`—indicates the identity of the concerned server
- `radiusAuthClientServerPortNumber`—indicates the port number of the concerned server
SNMP Traps

- `carAuthServerType`—indicates the type of the concerned server

The index of these three objects identifies the entry in `radiusAuthServerTable` and `carAccServerExtTable` which maintains the characteristics of the concerned server.

One should not rely on receiving this notification as an indication that all is well with the network. Several conditions, including a restart of the Prime Access Registrar server, could result in either multiple `carOtherAuthServerNotResponding` notifications being sent or in a `carOtherAuthServerResponding` notification not being sent. The NMS can query the `carAuthServerRunningState` in `carAuthServerExtTable` for the current running state of this server.

**carOtherAccServerNotResponding**

`carOtherAuthServerNotResponding` signifies that an accounting server is not responding to the requests sent from this server. This trap has three objects:

- `radiusAccServerAddress`—indicates the identity of the concerned server
- `radiusAccClientServerPortNumber`—indicates the port number of the concerned server
- `carAccServerType`—indicates the type of the concerned server

The index of these three objects identifies the entry in `radiusAuthServerTable` and `arAccServerExtTable` which maintains the characteristics of the concerned server.

One should not solely rely on this for server state. Several conditions, including the restart of the Prime Access Registrar server, could result in either multiple `carOtherAccServerNotResponding` notifications being sent or in a `carOtherAccServerResponding` notification not being sent. The NMS can query the `carAccServerRunningState` in `carAccServerExtTable` for current running state of this server.

**carOtherAccServerResponding**

`carOtherAccServerResponding` signifies that an accounting server that had previously sent a `not responding` message is now responding to requests from the Prime Access Registrar server. This trap has three objects:

- `radiusAccServerAddress`—indicates the identity of the concerned server
- `radiusAccClientServerPortNumber`—indicates the port number of the concerned server
- `carAccServerType`—indicates the type of the concerned server

The index of these three objects identifies the entry in `radiusAuthServerTable` and `arAccServerExtTable` which maintains the characteristics of the concerned server.

One should not rely on the reception of this notification as an indication that all is well with the network. Several conditions, including the restart of the Prime Access Registrar server, could result in either multiple `carOtherAccServerNotResponding` notifications being sent or in a `carOtherAccServerResponding` notification not being sent. The NMS can query the `carAccServerRunningState` in `carAccServerExtTable` for current running state of this server.

**carAccountingLoggingFailure**

`carAccountingLoggingFailure` signifies that this Prime Access Registrar server cannot record accounting packets locally. This trap has two objects:

- `carNotifAcctLogErrorReason`—indicates the reason packets cannot be recorded locally
carlicenseusage

carLicenseUsage signifies the percentage of transactions per second (TPS) or session License Usage.

TPS

The TPS trap is generated when the Prime Access Registrar server reaches license usage slabs namely 80%, 90%, 100%, and 110%. These traps are generated only once for every slab during the increasing steady state. Increasing steady state is a state when Prime Access Registrars’ incoming request rate shows 80% of the license usage over a period of 20 minutes. These traps will be regenerated only if a increasing steady state is observed after a decreasing steady state.

Concurrent Session

The concurrent session trap is generated when the Prime Access Registrar server reaches 80%. The incoming traffic slabs defined for trap generation are 80%, 90%, 100%, and 110% of the licensed Concurrent Sessions. These traps are generated once for every slab during the increasing steady state.

carSigtranLicenseUsage


carSigtranLicenseUsageTrap signifies the percentage of SIGTRAN transactions per second (SIGTRAN TPS) or SIGTRAN session License Usage.

carDiameterPeerDown


carDiameterPeerDown signifies that a Diameter peer is down. The identity of the peer is given by cdbpPeerIpAddress.

carDiameterPeerUp


carDiameterPeerUp signifies that a Diameter peer is up. The identity of the peer is given by cdbpPeerIpAddress.

Configuring Traps

The Prime Access Registrar SNMP implementation uses various configuration files to configure its applications.

This section contains the following topics:

- SNMP Configuration
- Configuring Trap Recipient
SNMP Configuration

A sample configuration file is available in /cisco-ar/ucd-snmp/share/snmp/snmpd.conf. This configuration file is used to configure SNMP query permissions and trap recipients.

Configuring Trap Recipient

The following example shows the default configuration that sets up trap recipients for SNMP versions v1 and v2c.

```
# ------------------------------------------------------------------------------
trapcommunity trapcom
trapsink zubat trapcom 162
trap2sink ponyta trapcom 162
#--------------------------------------------------------------------------------
```

**Note**
Most sites use a single NMS, not two as shown below.

```
# -------------------------------
trapcommunity trapcom
trapsink zubat trapcom 162
#-------------------------------
```

**Note**
trapsink is used in SNMP version 1; trap2sink is used in SNMP version 2.

trapcommunity defines the default community string to be used when sending traps. This command must appear prior to trapsink or trap2sink which use this community string.

trapsink and trap2sink are defined as follows:

```
trapsink  hostname  community  port
trap2sink hostname  community  port
```

Community String

A community string is used to authenticate the trap message sender (SNMP agent) to the trap recipient (SNMP management station). A community string is required in the list of trap receivers.
Enforcement of Licensing Models

This chapter describes the enforcement of transactions per second (TPS) based licensing and session based licensing models introduced in Cisco Prime Access Registrar (Prime Access Registrar).

In TPS based licensing model, the license is based on the number of transactions per second that are handled by the server. In session based licensing model, the license is managed based on the number of sessions that resides in Prime Access Registrar. During Prime Access Registrar startup, you can either load TPS based licensing or session based licensing, but not both at the same time.

This chapter contains the following sections:

- TPS Licensing Features
- Concurrent Session License Features

Note

The type of licensing will determine the applicable features and its corresponding enforcement.

TPS Licensing Features

The following are the features of TPS licensing:

- License will enable features but with restriction enforced on the TPS.
- TPS is the number of packets flowing into Prime Access Registrar. This is accounted by Prime Access Registrar irrespective of the feature being used.

This section contains the following topics:

- Enforcement Rules
- Notification Logs
- Notification - SNMP Traps
- TPS Logging Feature

Enforcement Rules

Any license enforcement is triggered only after Prime Access Registrar has observed increasing steady state in TPS. Increasing steady state is marked by the steady increase in incoming traffic (measured in TPS) beyond 80% of the licensed TPS for any 15 minutes of a 20 minute interval.
The following are the enforcement rules applied on reaching increasing steady state:

- When the incoming traffic (measured in TPS) is greater than 80% of the licensed TPS, SNMP Trap will be generated for the first time on reaching the increased steady state. The warning message on the current license usage is logged for every 5 minutes.

- When the incoming traffic (measured in TPS) is greater than 90% of the licensed TPS, SNMP Trap will be generated for the first time on reaching the increased steady state. Warning message on the current license usage is logged for every 5 minutes.

- When the incoming traffic (measured in TPS) is greater than 100% of the licensed TPS, SNMP Trap will be generated for the first time on reaching the increased steady state. Error message on the current license usage is logged for every 5 minutes.

- When the incoming traffic (measured in TPS) is greater than 110% of the licensed TPS, SNMP Trap will be generated for the first time on reaching the increased steady state.

Note

Steady state denotes continuous increase or decrease in the TPS within a given TPS range. For the purpose of enforcement of licensing in Prime Access Registrar, the range is always 80% and above. The enforcement begins after TPS reaches and is greater than 80% for a steady state of 20 minutes.

Notification Logs

A warning message is logged for every 5 minutes when the TPS count reaches an increased steady state, where, the TPS count is in the range of 80% to 100% of the licensed TPS.

An error message is logged for every 5 minutes when the TPS count reaches an increased steady state, where, the TPS count is in the range of 100% to 110% of the licensed TPS.

Notification - SNMP Traps

The carLicenseUsage traps are generated only once in an increasing phase. The incoming traffic slabs are defined as 80%, 90%, 100%, and 110% of the licensed TPS. When the incoming traffic slabs reaches an increasing steady state of 80% or above for the first time, the respective trap is generated for the slab.

If the TPS count drops below 80% of the licensed TPS for a steady state period of 20 minutes, Prime Access Registrar marks it as decreased or normal steady state. Traps will be regenerated again only if Prime Access Registrar observes a decreased steady state followed by an increased steady state of TPS falling under the slab (say 80%).
TPS Logging Feature

The properties in Advanced Object such as TPSSamplingPeriodInSecs, LogTPSActivity, TPSLogFilenamePrefix and TPSLogFileCount enable logging of TPS in the Prime Access Registrar server. TPS log file is located in /cisco-ar/logs. It creates one file per day to hold the TPS information for the day. The TPS samples are collected for every TPSSamplingPeriodInSecs. The file is updated only once for every 10* TPSSamplingPeriodInSecs. If there is no inflowing traffic, Prime Access Registrar logs zero TPS once for every 10* TPSSamplingPeriodInSecs. See Table 4-42 Advanced Object Properties for more information on TPSSamplingPeriodInSecs, LogTPSActivity, TPSLogFilenamePrefix, and TPSLogFileCount properties.

The following is the sample configuration of tps license:

```
/cisco-ar/bin/aregcmd -s
set /Radius/Advanced/LogTPSActivity TRUE
set /Radius/Advanced/TPSLogFilenamePrefix tps
set /Radius/Advanced/TPSLogFileCount 5
set /Radius/Advanced/TPSSamplingPeriodInSecs 30
save
```

The following is the sample output of the log file:

```
bash-3.00# tail -f tps-04-24-2014.csv
04-24-2014,18:36:30,2998,1000
04-24-2014,18:36:51,2997,1000
04-24-2014,18:37:11,996,408
04-24-2014,18:37:32,3532,1263
04-24-2014,18:37:53,2763,1000
04-24-2014,18:38:14,4749,1669
```

In the above example, the first numerical value (2998) in the log entry denotes the total TPS traffic and the second value (1000) denotes the SIGTRAN-M3UA traffic.

For TPS measurement in Prime Access Registrar server, you can run the TPS calculator script when you want to monitor the TPS during the peak period or run the TPS for 24 hours.

Concurrent Session License Features

In Concurrent Session based license, the licensing is done based on the number of sessions that resides in Prime Access Registrar.

During startup of the Prime Access Registrar, the default session manager must be enabled for RADIUS and the EnableStickySession must be set to TRUE for Diameter. This is applicable only for session based license.

The sticky sessions is enabled during the initialization of Prime Access Registrar for Diameter based license to track the session counts in the diameter service. The server level count is calculated by adding all the sessions maintained across all the session managers and the sticky sessions of all the diameter services in the server. This session count is used by licensing module for license enforcement. The session count is either increased or decreased based on the action performed.

This section contains the following topics:
- Sessions Enforcement Rules
- Notification Logs
• Notification - SNMP Traps
• Session Logging Feature

Sessions Enforcement Rules

The following are the enforcement rules applied on concurrent session based license:

• When the session count (measured in concurrent session) reaches 80% of the licensed sessions, SNMP Trap will be generated for the first time on reaching the increased steady state. The warning message on the current license usage is logged for every 5 minutes.

• When the session count (measured in concurrent session) reaches 90% of the licensed sessions, SNMP Trap will be generated for the first time on reaching the state. The warning message on the current license usage is logged for every 5 minutes.

• When the session count (measured in concurrent session) attains 100% of the licensed sessions, SNMP Trap will be generated for the first time on reaching the state. The error message on the current license usage is logged for every 5 minutes.

• When the session count (measured in concurrent session) attains 110% of the licensed sessions, SNMP Trap will be generated for the first time on reaching the state.

Note: The steady state period is not applicable for Concurrency Session based licensing.

Notification Logs

A warning message is logged for every 5 minutes when the session count reaches 80% and 90% of the licensed Concurrent Session.

An error message is logged when the session count reaches the range of 100% to 110% of the licensed Concurrent Session.
Notification - SNMP Traps

The `carLicenseUsage` trap is generated when the Prime Access Registrar server reaches 80%. The incoming traffic slabs defined for trap generation are 80%, 90%, 100%, and 110% of the licensed Concurrent Sessions. These traps are generated once for every slab during the increasing steady state.

Logging Feature is applicable for session based license as like TPS logging feature. The warning messages are displayed corresponding to session logging feature. See TPS Logging Feature, page 25-3 for more information.

Session Logging Feature

The properties in Advanced Object such as `SessionSamplingPeriodInSecs`, `LogSessionActivity`, `SessionLogFilenamePrefix` and `SessionLogFileCount` enable logging of session count in the Prime Access Registrar server. The session log file is located in `/cisco-ar/logs`. It creates one file per day to hold the session information for the day. The session samples are collected for every `SessionSamplingPeriodInSecs`. The file is updated only once for every 10* `SessionSamplingPeriodInSecs`. See Table 4-42 Advanced Object Properties for more information on `SessionSamplingPeriodInSecs`, `LogSessionActivity`, `SessionLogFilenamePrefix`, and `SessionLogFileCount` properties.

The following is a sample configuration of session license:

```
/cisco-ar/bin/aregcmd -s
set /Radius/Advanced/LogSessionActivity TRUE
set /Radius/Advanced/SessionLogFilenamePrefix sm
set /Radius/Advanced/SessionLogFileCount 5
set /Radius/Advanced/SessionSamplingPeriodInSecs 10
save
```

The following is the sample output of the log file:

```
bash-3.00# tail -f sm-08-09-2013.csv
08-09-2013, 5:18:52,73749
08-09-2013, 5:19:22,100001
08-09-2013, 5:19:52,100001
08-09-2013, 5:20:22,100001
08-09-2013, 5:20:52,100001
08-09-2013, 5:21:22,100001
08-09-2013, 5:21:52,100001
```
This chapter describes the Cisco Prime Access Registrar (Prime Access Registrar) shadow backup facility, which ensures a consistent snapshot of Prime Access Registrar’s database for backup purposes. Because the Prime Access Registrar’s database (called MCD) does a variety of memory caching, and might be active at any time, you cannot simply rely on doing system backups to protect the data in the database. At the time you run a system backup, there could be Prime Access Registrar operations in progress that cause the data copied to the system backup tape to be inconsistent and unusable as a replacement database.

To ensure a consistent backup, Prime Access Registrar uses a shadow backup facility. Once a day, at a configurable time, Prime Access Registrar suspends all activity to the database and takes a snapshot of the critical files. This snapshot is guaranteed to be a consistent view of the database, and it is preserved correctly on a system backup tape.

This chapter contains the following sections:

- Configuration
- Recovery
- mcdshadow Command Files

### Configuration

The only configuration for this facility is through a single entry in the system Registry at `$INSTALL/conf/car.conf` is the registry path to this item.

This entry is a string that represents the time-of-day at which the shadow backup is scheduled to occur (in 24 hour HH:MM format). The default is 12:45.

When you remove this entry or set it to an illegal value (for example, anything that does not begin with a digit), backups are suppressed.

### Command Line Utility

In addition to being available at a scheduled time of day, you can also force a shadow backup by using the `mcdshadow` utility located in the `$INSTALL/bin` directory. There are no command-line arguments. This might take a few minutes to complete as a full copy of the database is created.
Recovery

When it is necessary to use the shadow backup to recover data, either because the regular working database has been corrupted by a system crash, or because the disk on which it resides has become corrupted.

Recovering the data using shadow backup

To use the shadow backup to recover data:

Step 1  Stop all Prime Access Registrar servers.
Step 2  Make sure three files (mcdd.d01, mcdd.d02, and mcdd.d03) exist in the $INSTALL/data/db.bak directory.
Step 3  Copy the files into the $INSTALL/data/db directory. Do not move them because they might be needed again.
Step 4  Change directory to the $INSTALL/data/db directory.
          cd $INSTALL/data/db
Step 5  Rebuild the key files by entering the command:
          $INSTALL/bin/keybuild mcddb
This might take several minutes.
Step 6  As a safety check, run $INSTALL/bin/dbcheck mcddb (UNIX) to verify the integrity of the database. Note, you must be user root to run dbcheck.
          No errors should be detected.

mcdshadow Command Files

The mcdshadow command uses the files listed in Table 26-1.

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mcdd.dbd</td>
<td>Template file that describes the low-level data schema for the Raima runtime library.</td>
</tr>
<tr>
<td>mcdd.k01</td>
<td>Key files that contain the data that is redundant with the data files.</td>
</tr>
<tr>
<td>mcdd.k02</td>
<td>Prime Access Registrar does not back up these files because they can be completely rebuilt with the keybuild command.</td>
</tr>
<tr>
<td>mcdd.k03</td>
<td></td>
</tr>
<tr>
<td>mcdd.d01</td>
<td>Data files that contain the backup.</td>
</tr>
<tr>
<td>mcdd.d02</td>
<td></td>
</tr>
<tr>
<td>mcdd.d03</td>
<td></td>
</tr>
<tr>
<td>mcdConfig.txt</td>
<td>Text file from which Prime Access Registrar configures the initial at-install-time database.</td>
</tr>
</tbody>
</table>
### Table 26-1  mcdshadow Files (continued)

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mcdschema.txt</td>
<td>Text file that contains a version number denoting the level of the schema contained in the dbd file. Prime Access Registrar will not attempt to open the database unless the number in this file matches a constant that is hard-coded in the libraries. If the result of the mcdshadow command (which uses copies of the data files) is divorced from its original mcdschema.txt, you will not be able to run Prime Access Registrar.</td>
</tr>
<tr>
<td>vista.taf</td>
<td>Working files used by the Raima runtime library to ensure transactional integrity.</td>
</tr>
<tr>
<td>vista.tcf</td>
<td></td>
</tr>
<tr>
<td>vista.tjf</td>
<td></td>
</tr>
</tbody>
</table>
Using the REX Accounting Script

This chapter describes how to use the REX Accounting Script (RAS). The RAS writes RADIUS Accounting requests to a local, flat file and is included as an option for Cisco Prime Access Registrar (Prime Access Registrar). It is designed to be attached to a Prime Access Registrar IncomingScript or OutgoingScript point. When used in conjunction with the Prime Access Registrar built-in proxy support, the server will concurrently store a local copy of an Accounting request and proxy another copy to another RADIUS server.

Note

Unless you require log rotation at an exact time or when the accounting log reaches a specific file size, we recommend that you use service grouping to log and proxy accounting packets.

RAS can be attached to more than one Prime Access Registrar extension point. For example, in a dial-up resale scenario, you might configure Prime Access Registrar to proxy Accounting requests to many different Remote Servers (by realm). For some subset of those, you might want to keep a local copy of the Accounting requests. In this case, RAS could be installed as the IncomingScript on just the Services for which a local copy is desired.

Note

Also included is the DropAcctOnOff Script. This script causes Prime Access Registrar to drop all Accounting-Requests with an Acct-Status-Type of Accounting-On or Accounting-Off.

This chapter contains the following sections:

- Building and Installing the REX Accounting Script
- Configuring the Rex Accounting Script
- Specifying REX Accounting Script Options

Building and Installing the REX Accounting Script

The RAS writes RADIUS Accounting requests to a local, flat file and is included as an option for Prime Access Registrar. It is designed to be attached to a Prime Access Registrar IncomingScript or OutgoingScript point.
Building and Installing the REX Accounting Script

To build and install RAS:

**Step 1** Change directory to `$INSTALL/examples/rexaacctscript`.

**Step 2** Modify the `Makefile` to ensure the `AR_INSTALL_DIR` variable points to the directory where the Prime Access Registrar software was installed, and then choose a compiler (`gcc` or SUNPro `CC`).

**Step 3** From the command line prompt, enter:

```
host% make
```

**Step 4** Log in as user `root`.

**Step 5** From the command line prompt, enter:

```
host# make install
```

Configuring the Rex Accounting Script

RAS can be attached to more than one Prime Access Registrar extension point. For example, in a dial-up resale scenario, you might configure Prime Access Registrar to proxy Accounting requests to many different Remote Servers (by realm).

**Step 1** Start the Prime Access Registrar `aregcmd` configuration utility and login:

```
> $INSTALL/usrbin/aregcmd -C localhost -N admin -P aicuser
Access Registrar Configuration Utility Version 1.3
Copyright (C) 1995-2013 by Cisco Systems, Inc. All rights reserved.
Logging in to localhost
[ //localhost ]
    LicenseKey = xxxx-xxxx-xxxx-xxxx
    Radius/
    Administrators/
Server 'Radius' is Running, its health is 10 out of 10
--> 
```

**Step 2** Using `aregcmd`, create a new Prime Access Registrar Script object:

```
--> cd /Radius/Scripts
[ //localhost/Radius/Scripts ]
    Entries 1 to 20 from 39 total entries
    Current filter: <all>
    ACMEOutgoingScript/
    AscendIncomingScript/
    <... other output deleted...>

--> add LocalAccounting
    Added LocalAccounting
```

```plaintext
---> cd LocalAccounting
[ //localhost/Radius/Scripts/LocalAccounting ]
  Name = LocalAccounting
  Description =
  Language =
  Filename =
  EntryPoint =
  InitEntryPoint =
  InitEntryPointArgs =

  set Desc "Log Accounting requests to local file"
  Set Description "Log Accounting requests to local file"

  --> set lang REX
  Set Language REX

  --> set filename libRexAcctScript.so
  Set Filename libRexAcctScript.so

  --> set entry RexAccountingScript
  Set EntryPoint RexAccountingScript

  --> set initentrypoint InitRexAccountingScript
  Set InitEntryPoint InitRexAccountingScript

  --> set initentrypointargs "-f Accounting -t 1:15"
  Set InitEntryPointArgs "-f Accounting -t 1:15"

  --> ls
  [ //localhost/Radius/Scripts/LocalAccounting ]
    Name = LocalAccounting
    Description = "Log Accounting requests to local file"
    Language = REX
    Filename = libRexAcctScript.so
    EntryPoint = RexAccountingScript
    InitEntryPoint = InitRexAccountingScript
    InitEntryPointArgs = "-f Accounting -t 1:15"

  -->
```

Step 4  Using aregcmd, attach the new Prime Access Registrar Script object to the appropriate Prime Access Registrar Scripting point. See Chapter 4, “Cisco Prime Access Registrar Server Objects,” for more details.

```plaintext
---> set /radius/IncomingScript LocalAccounting
Set /Radius/IncomingScript LocalAccounting
```

Step 5  Using aregcmd, save the configuration modifications:

```plaintext
---> save
Validating //localhost...
Saving //localhost...
```

Step 6  Using aregcmd, reload the server:
### Specifying REX Accounting Script Options

The REX Accounting Script supports the options shown in Table 27-1.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f &lt;filename&gt;</td>
<td>Required. Specify the name of the output file.</td>
</tr>
<tr>
<td>-t <a href="">HH:MM[:SS]</a></td>
<td>Specify a time of day to roll the output file. Note, this is time on the 24-hour clock, for example, 00:05 = 12:05am, 13:30 = 1:30pm. This option can not be used with the -i option.</td>
</tr>
<tr>
<td>-i &lt;seconds&gt;</td>
<td>Specify the number of seconds between rolling the output file, beginning at start-up. This option can not be used with the -t option.</td>
</tr>
<tr>
<td>-s &lt;size&gt;[kmlg]</td>
<td>Specify the maximum size for an output file. When the file reaches this size, it will be rolled. When specifying the &lt;size&gt; option, a &lt;unit&gt; can be included. When a &lt;unit&gt; is not included, the &lt;size&gt; is in bytes. Note, do not use a space character between the &lt;size&gt; and &lt;unit&gt; options. &lt;unit&gt; can be either: k = 1K, m = 1Meg, g = 1Gig.</td>
</tr>
<tr>
<td>-g</td>
<td>Use GMT when writing the date/time in the Accounting output file for each record (default is local time).</td>
</tr>
<tr>
<td>-G</td>
<td>Use GMT when naming rolled output files (default is local time).</td>
</tr>
<tr>
<td>-A</td>
<td>Process all packets, not just Accounting-Requests.</td>
</tr>
<tr>
<td>-I</td>
<td>Ignore errors when processing packets, always return successfully.</td>
</tr>
<tr>
<td>-a &lt;buffer-count&gt;</td>
<td>Pre-allocate this many Accounting buffers to improve performance.</td>
</tr>
<tr>
<td>-T &lt;trace-level&gt;</td>
<td>Set the trace level. This trace info appears in the output file (as its written by the background thread which no longer has a packet to use for logging or tracing.)</td>
</tr>
<tr>
<td>-O</td>
<td>Call another REX extension before calling the RexAcctScript.</td>
</tr>
<tr>
<td>&lt;script-description&gt;</td>
<td>Call another REX extension after calling the RexAcctScript.</td>
</tr>
</tbody>
</table>
Example Script Object

This is an example of what a Prime Access Registrar Script object using RAS might look like when viewed in the Prime Access Registrar configuration utility, aregcmd:

[ //localhost/Radius/Scripts/REX-Accounting-Script ]
Name = REX-Accounting-Script
Description =
Language = REX
Filename = librexacctscript.so
EntryPoint = RexAccountingScript
InitEntryPoint = InitRexAccountingScript
InitEntryPointArgs = "-f Accounting -t 16:20 -s 100k -o libRexAcctScript.so:DropAcctOnOff"

This example causes RAS to write to a file called Accounting.log (in the logs directory of the installation tree). The file rolls every day at 4:20pm (local time), as well as whenever it grows larger than 100k in size. RAS also runs the DropAcctOnOff script against every packet, after it has processed the packet.
Logging Syslog Messages

Logging messages via syslog provides centralized error reporting for Cisco Prime Access Registrar (Prime Access Registrar). Local logging and syslog logging can be turned on or off at any time by modifying the control flags in the $INSTALLPATH/conf/car.conf file.

Logging syslog messages requires a UNIX host running a syslog daemon as a receiver for Prime Access Registrar messages. Prime Access Registrar and the syslog daemon can be running on the same host or different hosts.

This chapter contains the following sections:

- Syslog Messages
- Configuring Message Logging
- Configuring Message Logging
- Changing Log Directory
- Configuring Syslog Daemon (syslogd)
- Managing the Syslog File
- Server Up/Down Status Change Logging

Syslog Messages

Messages sent to the following logs will be forwarded to syslog server in a slightly different format. The logs are:

- aregcmd_log
- config_mcd_[1..n]_log
- name_radius_[1..n]_log
- agent_server_[1..n]_log

Messages less than 1024 bytes in length display in the following format:

`MMM DD hh:mm:ss hostname %Prime AR-[severity]-[mnemonic]: [#n], [System|Server]: message_description`

Where:

- **MMM DD** is the month and date that the message is received by the syslog server.
- **hh:mm:ss** is the arrival time of the message.
- **hostname** is the name of the syslog server.
**severity** is one of the following levels:

0 - emergency
1 - alert
2 - critical
3 - error
4 - warning
5 - notification
6 - informational
7 - debugging

**mnemonic** can be aregcmd, name_radius, agent_server and config_mcd for the identification of Prime Access Registrar-relative subsystems.

**#n** is the id for the components: name_radius, agent_server, and config_mcd

**message_description** provides detailed information of the message.

Messages greater than 1024 bytes in length display in multiple lines. At the end of each 1024 bytes line, three dots indicate a continuation of the message as follows:

```
MMM DD hh:mm:ss hostname %Prime AR-[severity]-[mnemonic]: [#n], [System|Server]:
message_description: Configuration: text and more message text and more message text
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message...
```

The continuation of a message begins with three dots as follows:

```
MMM DD hh:mm:ss hostname %Prime AR-[severity]-[mnemonic]: [#n], [System|Server]:
message_description: Configuration: ... text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message
and more message text and more message text and more message text and more message

Example 1

May 19 14:28:44 dwlau-ultra2.cisco.com

%Prime AR-3-name_radius: #1, System: Remote LDAP Server.Unable to bind.

Example 2

May 19 14:28:45 dwlau-ultra2.cisco.com

%Prime AR-6-name_radius: #1, Server: Stopping server
Configuring Message Logging

To enable syslog logging in Linux, you must modify the syslog file in the /etc/sysconfig directory. The following is the default syslog file.

```
# Options to syslogd
# -m 0 disables 'MARK' messages.
# -r enables logging from remote machines
# -x disables DNS lookups on messages recieved with -r
# See syslogd(8) for more details
SYSLOGD_OPTIONS="-m 0"
# Options to klogd
# -2 prints all kernel oops messages twice; once for klogd to decode, and
# once for processing with 'ksymoops'
# -x disables all klogd processing of oops messages entirely
# See klogd(8) for more details
KLOGD_OPTIONS="-x"
```

To enable logging of syslog messages, you must enable the syslog daemon to listen on port 514 by adding the -r flag to the SYSLOGD_OPTIONS line as follows:

```
SYSLOGD_OPTIONS="-m 0 -r"
```

Changing Log Directory

You can change the directory where local log messages are stored by adding the following line in the $INSTALLPATH/conf/car.conf file.

```
LOGDIR full_path
```

Where `full_path` is a full path to the directory where you want to store the log messages. For example, to store all system logs in /var/log/AICar1, add the following line in the $INSTALLPATH/conf/car.conf file:

```
LOGDIR /var/log/AICar1
```

You must first stop the Prime Access Registrar server prior to changing the car.conf file. After changing the car.conf file, copy all existing log files to the new directory, then restart the server.

---

**Note**

Specifying a path for local logging does not affect the storage location of syslog messages.
Configuring Syslog Daemon (syslogd)

You must specify the facility from which syslogd will receive messages and the file into which the messages will be deposited.

In the syslog server's /etc/syslog.conf file, the following line might be needed.

```
localn.info <tab> <tab> <tab> /var/log/filename.log
```

**Note**  Use at least one <tab> as a field separator.

Where:

- **localn**—is the facility being used for syslogd; \( n \) must be a value from 0-7 and match the FACILITY_LOCAL_NUMBER used in Prime Access Registrar's car.conf file.
- **/var/log/**—is the path to the file that stores syslogd messages.
- **filename.log**—is the file that stores syslogd messages. You can give this file a name of your choice.

Creating a Syslog Log File

To create a syslog log file:

**Step 1**  Log in as user **root**.

**Step 2**  Enter the following command, where **filename.log** is a name you choose.

```
touch filename.log
```

**Step 3**  Change permissions on the syslog log file by entering the following:

```
chmod 664 filename.log
```

Restarting a syslog daemon

To restart the syslog daemon:

log in as user **root** and enter the following commands:

```
/etc/init.d/syslog stop
/etc/init.d/syslog start
```

Managing the Syslog File

Left unmanaged, the syslog file will grow in size over time and eventually fill all available disk space in its partition. Prime Access Registrar writes log files and session data (to persist user sessions) in the same disk partition where Prime Access Registrar is installed.

In normal operation, log files consume a large amount of disk space. If log files are not managed regularly, Prime Access Registrar might not have sufficient disk space to write session data. To avoid this, you should move the Prime Access Registrar log files directory to a different disk partition than the one where Prime Access Registrar writes session data, as described in Changing Log Directory.
Using a cron Program to Manage the Syslog Files

We recommend that you use the cron program to manage the syslog files.

The following example crontab file performs a weekly archival of the existing syslog file (named ar_syslog.log in this example). This scheme keeps the previous two week’s worth of syslog files.

```
# At 02:01am on Sundays:
# Move a weeks worth of 'ar_syslog.log' log messages to 'ar_syslog.log.1'.
# If there was a 'ar_syslog.log.1' move it to 'ar_syslog.log.2'.
# If there was a 'ar_syslog.log.2' then it is lost.
01 02 * * 0 cd /var/log;
if [ -f ar_syslog.log ];
then if [ -f ar_syslog.log.1 ];
then /bin/mv ar_syslog.log.1 ar_syslog.log.2;
fi;
/usr/bin/cp ar_syslog.log ar_syslog.log.1;
>ar_syslog.log;
fi
```

Note

Consider using move (mv) or copy (cp) commands to store the previous week’s syslog files in a different disk partition to reserve space for the current syslog file.

Using a cron Program to Manage the Syslog Files

To add this crontab segment to the existing cron facility in /usr/spool/cron/crontabs directory, complete the following steps at the syslog server console:

**Step 1** Log in as user root.

**Step 2** Enter the following command:

```
crontab -e
```

Server Up/Down Status Change Logging

Prime Access Registrar supports RADIUS server up/down detection and logging. The information messages are saved in the $INSTALL/logs/name_radius_1_log file where $INSTALL is the Prime Access Registrar installation directory. Each message consists of a header and a message description.

Header Formats

The format of a header entry is:

```
mm/dd/yyyy HH:MM:SS name/radius/n Error Server 0
```
Example Log Messages

Following are the descriptions and types of messages that can be found within the `<AR_install_dir>/logs/name_radius_1_log` file:

1. Prime Access Registrar detects a Remote Server when it responds for the first time or after it is reentered into Prime Access Registrar’s server pool for retry. The format of the message is:
   
   Remote Server `<hostname>` ( `<ipaddress>`: `<port>`) is UP!
   
   The following is an example header and message:
   
   10/12/2013 17:56:32 name/radius/1 Error Server 0 Remote Server dave-ultra (171.69.127.99:1812) is UP!

   Prime Access Registrar detects the Remote Server is not responding to its request. The format of the message is:

   Remote Server `<hostname>` ( `<ipaddress>`: `<port>`) is DOWN!

   The following is an example header and message:

   10/12/2013 17:57:12 name/radius/1 Error Server 0 Remote server dave-ultra (171.69.127.99:1812) is DOWN!

2. Prime Access Registrar receives no response from the Remote Server after the server is reentered into Prime Access Registrar’s server pool for retry. The format of the message is:

   Remote Server `<hostname>` ( `<ipaddress>`: `<port>`) remains DOWN!

   The following is an example header and message:

   10/12/2013 17:56:32 name/radius/1 Error Server 0 Remote server dave-ultra (171.69.127.99:1812) remains DOWN!

3. The Remote Server is responding to the first retry but not the initial request. The format of the message is:

   Remote Server `<hostname>` ( `<ipaddress>`: `<port>`) is UP but slow!

   The following is an example header and message:

   10/12/2013 17:56:32 name/radius/1 Error Server 0 Remote server dave-ultra (171.69.127.99:1812) is UP but slow!

4. The Remote Server is responding to the second retry request but not the initial request or the first retry request. The format of the message is:

   Remote Server `<hostname>` ( `<ipaddress>`: `<port>`) is UP but very slow!

   The following is an example header and message:

   10/12/2013 17:56:32 name/radius/1 Error Server 0 Remote server dave-ultra (171.69.127.99:1812) is UP but very slow!

5. The Remote Server has been marked inactive and is being put back into Prime Access Registrar’s server pool for later use. The format of the message is:

   Remote Server `<hostname>` ( `<ipaddress>`: `<port>`) is being reactivated for later use.

   The following is an example header and message:

   10/12/2013 17:56:32 name/radius/1 Error Server 0 Remote server dave-ultra (209.165.200.224:1812) is being reactivated for later use.
Troubleshooting Cisco Prime Access Registrar

This chapter provides information about techniques used when troubleshooting Cisco Access Registrar (Prime Access Registrar) and highlights common problems.

This chapter contains the following sections:
- Gathering Basic Information
- Troubleshooting Quick Checks
- aregcmd and Cisco Prime Access Registrar Configuration
- RADIUS Request Processing
- Other Troubleshooting Techniques and Resources
- Checking Prime Access Registrar Server Health Status

Gathering Basic Information

Table 29-1 lists UNIX commands that provide basic and essential information to help you understand the Prime Access Registrar installation environment.

<table>
<thead>
<tr>
<th>UNIX Command</th>
<th>Information Returned</th>
</tr>
</thead>
<tbody>
<tr>
<td>/usr/bin/uname -r</td>
<td>Release level</td>
</tr>
<tr>
<td>/usr/bin/uname -i</td>
<td>Machine hardware name</td>
</tr>
<tr>
<td>/usr/bin/uname -v</td>
<td>OS version</td>
</tr>
<tr>
<td>/usr/bin/uname -a</td>
<td>All system information including hostname, operating system type and release, machine model and type</td>
</tr>
<tr>
<td>/usr/sbin/prtconf</td>
<td>System configuration information including memory capacity, machine type, and peripheral equipment</td>
</tr>
<tr>
<td>/usr/sbin/df -k</td>
<td>File system disk space usage including partitions, capacity, and space used</td>
</tr>
<tr>
<td>/usr/bin/ps -ef</td>
<td>Currently running processes</td>
</tr>
</tbody>
</table>
Troubleshooting Quick Checks

Many of the most common problems can be diagnosed by doing the following:

- Check disk space
- Check for resource conflicts
- Check the Prime Access Registrar log files

Disk Space

Running out of disk space can cause a number of problems including:

- Failure to process RADIUS requests
- Parts of the Prime Access Registrar configuration disappearing in aregcmd
- Failure to log into aregcmd

Check that the Prime Access Registrar installation partition ($INSTALL) and /tmp are not at capacity.

Resource Conflicts

Resource conflicts are a common reason for the Cisco Prime Access Registrar server failing to start. The most common resource conflicts are the following:

- Cisco Network Registrar is running on the Prime Access Registrar server
- Another application is also using ports 1812 and 1813
- A network management application is using the Sun SNMP Agent

No Co-Existence With Cisco Network Registrar

Cisco Network Registrar cannot coexist on a machine running Prime Access Registrar for this reason. You can determine if CNR is running by entering the following command line in a terminal window:

```
pkginfo | grep -i “network registrar”
```
Port Conflicts

The default ports used by the Prime Access Registrar server are ports 1812 and 1813. You should check to determine that no other applications are listening on the same ports as Prime Access Registrar.

You can check to see which TCP ports are in use by entering the following command line:

```
netstat -aP tcp
```

You can check to see which UDP ports are in use by entering the following command line:

```
netstat -aP udp
```

**Note**

If you configure the Prime Access Registrar server to use ports other than the default, you will have to specifically add those ports if you want to use them.

Cisco Prime Access Registrar Log Files

Examining the Prime Access Registrar log files can help you diagnose most Prime Access Registrar issues. By default, the Prime Access Registrar log files are located in `/opt/CSCOar/logs`. Table 29-2 lists the Prime Access Registrar log files and the information stored in each log.

<table>
<thead>
<tr>
<th>Log File</th>
<th>Information Recorded</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent_server_1_log</td>
<td>Log of the server agent process</td>
</tr>
<tr>
<td>ar-status</td>
<td>Log of Prime Access Registrar stop and start using the arserver utility</td>
</tr>
<tr>
<td>aregcmd_log</td>
<td>Log of commands executed in aregcmd (very useful for tracing the steps that took place before a problem occurred)</td>
</tr>
<tr>
<td>config_mcd_1_log</td>
<td>Log of the mcd internal database</td>
</tr>
<tr>
<td>name_radius_1_log</td>
<td>Log of the radius server process</td>
</tr>
<tr>
<td>name_radius_1_trace</td>
<td>Debugging output of RADIUS request processing (only generated when the trace level, set in aregcmd, is greater than zero)</td>
</tr>
<tr>
<td>m3ua</td>
<td>System logs related to stack including m3ua client and stack manager; and radius communication logs related to m3ua stack. These are typically low level debug logs</td>
</tr>
</tbody>
</table>

Modifying File Sizes for Agent Server and MCD Server Logs

The two parameters added to the car.conf file under $BASEDIR/conf affect the agent_server_logs and config_mcd_server_logs logs files:

- AGENT_SERVER_LOG_SIZE (10 MB by default)
- AGENT_SERVER_LOG_FILES (2 by default)
You will find these new parameters at the beginning of the `car.conf` file. When the log file size reaches the value set in `AGENT_SERVER_LOG_SIZE`, a rollover of the `agent_server_log_file` occurs. The value set in `AGENT_SERVER_LOG_FILES` specifies the number of log files to be created.

**Using xtail to Monitor Log File Activity**

A useful way of monitoring all of the log files is to run `xtail`, a utility provided with Prime Access Registrar. The `xtail` program monitors one or more files and displays all data written to a file since command invocation.

Run `xtail` in a dedicated terminal window. It is very useful for monitoring multiple logfiles simultaneously, such as with a command line like the following:

```bash
xtail $INSTALL/logs/*
```

**Note** Cisco AR 4.1.5 and later include the millisecond field in the logs’ timestamp.

**Modifying the Trace Level**

By modifying the trace level, you can gather more detailed information in the log files about what is happening in the Prime Access Registrar server. There are five different trace levels. Each higher trace level also includes the information logged using lower trace levels. The different trace levels provide the following information:

- **Level 0**—No tracing occurs
- **Level 1**—Indicates when a packet is sent or received and when a status change occurs in a remote server (RADIUS Proxy and LDAP)
- **Level 2**—Information includes the following:
  - Which services and session managers are used to process
  - Which client and vendor objects are being used to process a packet
  - More details about remote servers (RADIUS Proxy and LDAP), packet transmission, and timeouts
  - Details about poorly-formed packets.
- **Level 3**—Information includes the following:
  - Tracing of errors in Tcl scripts when referencing invalid RADIUS attributes
  - Which scripts have been run
  - Details about local userlist processing
- **Level 4**—Information includes the following:
  - Advanced duplication detection processing
  - Details about creating, updating, and deleting sessions
  - Tracing of all APIs called during the running of a script
- **Level 5**—Provides information about policy engine operations
Installation and Server Process Start-up

The installation process installs the Prime Access Registrar software to the specified installation directory and then starts the server processes. This process rarely fails but the following checks should always be performed:

- Ensure that there is an **installation success message** at the end of the `pkgadd` dialog, otherwise check the dialog for the problem.
- Follow the installation instructions carefully especially when performing an upgrade. For example, when upgrading to 1.6R1, 1.6R2, or 1.6R3, a post-installation upgrade script needs to be run.
- Pay attention to the information included in README files.

At the end of a successful installation, `arstatus` should show the following four server processes:

```bash
> $INSTALL/usrbin/arstatus
AR RADIUS server running    (pid: 6285)
AR MCD lock manager running (pid: 6284)
AR MCD server running       (pid: 6283)
AR Server Agent running     (pid: 6277)
```

If any of the above processes are not displayed, check the log file of the failed process to determine the reason. The MCD processes might fail to start if Cisco Network Registrar is installed on the same machine.

The manual method of starting and stopping the Prime Access Registrar processes is using the `arserver` utility.

- To start Prime Access Registrar processes: `arserver start`
- To stop Prime Access Registrar processes: `arserver stop`
- To restart Prime Access Registrar processes: `arserver restart`

aregcmd and Cisco Prime Access Registrar Configuration

While troubleshooting, you should always use the `aregcmd` command trace to turn on tracing. With tracing active, Prime Access Registrar generates debugging output to the log file `name_radius_1_trace`. The syntax is:

```bash
trace [<server>] [<level>]
```

When you do not specify a server, Prime Access Registrar sets the trace level for all servers in the current cluster. When you do not specify a trace level, the currently set level is used. The default trace level is 0.

Running and Stopped States

Prime Access Registrar can be in two states, running or stopped. In either state, all four Prime Access Registrar processes remain running. The state of Prime Access Registrar will be displayed when logging into `aregcmd` or by using the `aregcmd status` command:

```bash
status
```

```
Server 'Radius' is Running, its health is 10 out of 10
```
The **start** and **stop** commands allow Prime Access Registrar to move between states. **Reload** is equivalent to a **stop** followed by a **start** if Prime Access Registrar is already running, and just a **start** if it is already stopped.

```
stop

Stopping Server 'Radius'...
Server 'Radius' is Stopped
```

```
start

Starting Server 'Radius'...
Server 'Radius' is Running, its health is 10 out of 10
```

```
reload

Reloading Server 'Radius'...
Server 'Radius' is Running, its health is 10 out of 10
```

During the transition from running to stopped, Prime Access Registrar stops processing new RADIUS requests and releases resources such memory, network and database connections and open files.

During the transition from stopped to running, Prime Access Registrar reverses this process by opening a connection with its internal database, reading configuration data, claiming memory, establishing network connections, opening files, and initializing scripts. During this transition, problems can occur. Prime Access Registrar might fail to start and display the following:

```
reload

Reloading Server 'Radius'...
310 Command failed
```

Prime Access Registrar failed to move from stopped state to running:

```
status

Server 'Radius' is Stopped
```

This might occur for a number of reasons including the following:

- An invalid configuration
- Insufficient memory
- Listening ports already in use by another application
- Unable to open files
- Unable to initialize scripts

Check the **name_radius_1_log** file for the one of these indications.
Chapter 29      Troubleshooting Cisco Prime Access Registrar

RADIUS Request Processing

The main technique for troubleshooting RADIUS request processing in Prime Access Registrar is to examine the `name_radius_1_trace` log file with the trace level set to 5. Most issues are fairly self-explanatory. Some issues that can arise are as follows:

- Prime Access Registrar has marked a remote server as `down`
- A resource manager has run out of resources (for example, user or group session limit has been reached or no more IP addresses are available)
- A configuration error (such as an accounting service not being set)
- A run time error in a script

Some issues are not immediately evident from the log files though, such as the following:

- Failure to save or reload Prime Access Registrar after a configuration change
- Prime Access Registrar is not listening on the correct UDP ports for RADIUS requests

Other Troubleshooting Techniques and Resources

aregcmd Stats Command

The `aregcmd` command `stats` provides statistics on request processing.

```bash
---> stats
Global Statistics for Radius:
    serverStartTime = Tue Oct  2 10:28:02 2013
    serverResetTime = Tue Oct  2 20:25:12 2013
    serverState = Running
    totalPacketsInPool = 1024
    totalPacketsReceived = 0
    totalPacketsSent = 0
    totalRequests = 0
    totalResponses = 0
    totalAccessRequests = 0
    totalAccessAccepts = 0
    totalAccessChallenges = 0
    totalAccessRejects = 0
    totalAccessResponses = 0
    totalAccountingRequests = 0
    totalAccountingResponses = 0
    totalStatusServerRequests = 0
    totalAscendIPAAllocateRequests = 0
    totalAscendIPAAllocateResponses = 0
    totalAscendIPAReleaseRequests = 0
    totalAscendIPAReleaseResponses = 0
    totalUSRNASRebootRequests = 0
    totalUSRNASRebootResponses = 0
    totalUSRResourceFreeRequests = 0
    totalUSRResourceFreeResponses = 0
```
Checking Prime Access Registrar Server Health Status

To check the server’s health, use the aregcmd command status. The following issues decrement the server’s health:

- Multiple occurrences of Access-Request rejection

Core Files

A core file in the Prime Access Registrar installation directory is an indication that Prime Access Registrar has crashed and restarted. Check that the radius server process generated the core file using the UNIX file command:

```
> file core
core:          ELF 32-bit MSB core file SPARC Version 1, from 'radius'
```

Check the timestamp on the core file and look for corresponding log messages in the name_radius_1_log file in INSTALL/logs. The word assertion commonly appears in core messages. Try to establish what caused the problem and contact Cisco TAC.

radclient

The Prime Access Registrar package provides a utility called radclient that allows RADIUS requests to be generated. Use radclient to test configurations and troubleshoot problems.

Cisco Prime Access Registrar Replication

For more information about using Prime Access Registrar replication, see Chapter 12, “Using Replication.”

Checking Prime Access Registrar Server Health Status

Global Statistics for Radius:
serverStartTime = Sat Dec 7 07:37:52 2013
serverResetTime = Sat Dec 7 07:37:52 2013
serverState = Running
cdbpLocalStatsTotalUpTime = 1486 seconds
cdbpLocalResetTime = 1486 seconds
cdbpLocalStatsTotalPacketsIn = 30
cdbpLocalStatsTotalPacketsOut = 30
cdbpLocalStatsTotalPacketsInUse = 0

totalUSRQueryResourceRequests = 0
totalUSRQueryResourceResponses = 0
totalUSRQueryReclaimRequests = 0
totalUSRQueryReclaimResponses = 0
totalPacketsInUse = 0
totalPacketsDrained = 0
totalPacketsDropped = 0
totalPayloadDecryptionFailures = 0

Checking Prime Access Registrar Server Health Status

Global Statistics for Radius:
serverStartTime = Sat Dec 7 07:37:52 2013
serverResetTime = Sat Dec 7 07:37:52 2013
serverState = Running
cdbpLocalStatsTotalUpTime = 1486 seconds
cdbpLocalResetTime = 1486 seconds
cdbpLocalStatsTotalPacketsIn = 30
cdbpLocalStatsTotalPacketsOut = 30
cdbpLocalStatsTotalPacketsInUse = 0
One of the parameters in the calculation of the Prime Access Registrar server’s health is the percentage of responses to Access-Accepts that are rejections. In a healthy environment, the rejection percentage will be fairly low. An extremely high percentage of rejections could be an indication of a Denial of Service attack.

- Configuration errors
- Running out of memory
- Errors reading from the network
- Dropping packets that cannot be read (because the server ran out of memory)
- Errors writing to the network.

Prime Access Registrar logs all of these conditions. Sending multiple successful responses to any packet, increments the server’s health.
Cisco Prime Access Registrar Tcl, REX, and Java Dictionaries

This appendix describes the Tcl and REX dictionaries that are used when writing Incoming or Outgoing scripts.

A dictionary is a data structure that contains key/value pairs. Two types of dictionaries exist: the Attribute dictionaries (used by the Request and Response dictionaries), and the Environment dictionary.

This section contains the dictionaries you reference when writing a Tcl script and the dictionaries you reference when you write a script using the shared libraries (REX—RADIUS EXtension).

This appendix section also describes the following Java attribute dictionary:

- Tcl Attribute Dictionaries
- REX Attribute Dictionary
- Java Attribute Dictionary

Tcl Attribute Dictionaries

An Attribute dictionary is a dictionary in which the keys are constrained to be the names of attributes as defined in the Prime Access Registrar server configuration, and the values are the string representation of the legal values for that particular attribute. For example, IP addresses are specified by the dotted-decimal string representation of the address, and enumerated values are specified by the name of the enumeration. This means numbers are specified by the string representation of the number.

Attribute dictionaries have the unusual feature that there can be more than one instance of a particular key in the dictionary. These instances are ordered, with the first instance at index zero. Some of the methods of an Attribute dictionary allow an index to be specified to indicate a particular instance or position in the list of instances to be referenced. This section contains the following topics:

- Attribute Dictionary Methods
- Tcl Environment Dictionary

Attribute Dictionary Methods

Attribute dictionaries use active commands, called methods, that allow you to change and access the values in the dictionaries. Table A-1 lists all of the methods you can use with the Request and Response dictionaries.
### Table A-1  Tcl Attribute Dictionary Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>addProfile</td>
<td>$dict addProfile &lt;profile&gt; [ &lt;mode&gt; ]</td>
<td>Copies all of the attributes in the profile &lt;profile&gt; into the dictionary. Note, &lt;profile&gt; must be the name of one of the profiles listed in the server configuration. When &lt;mode&gt; is not provided or when &lt;mode&gt; equals the special value REPLACE, any duplicate instances of the attributes in the dictionary are replaced with the attribute from &lt;profile&gt;. When &lt;mode&gt; is provided and equals the special value APPEND, new instances of the attributes are appended to the attributes already in the dictionary. When &lt;mode&gt; is provided and equals the special value AUGMENT, only add the attribute when it does not already exist.</td>
</tr>
<tr>
<td>clear</td>
<td>$dict clear</td>
<td>Removes all entries from the dictionary.</td>
</tr>
<tr>
<td>containsKey</td>
<td>$dict containsKey &lt;attribute&gt;</td>
<td>Returns 1 when the dictionary contains the attribute &lt;attribute&gt;, otherwise returns 0.</td>
</tr>
<tr>
<td>firstKey</td>
<td>$dict firstKey</td>
<td>Returns the name of the first attribute in the dictionary. Note, the attributes are not stored in a sorted order of name.</td>
</tr>
<tr>
<td>get</td>
<td>$dict get &lt;attribute&gt; [ &lt;index&gt; [ bMore]]</td>
<td>Returns the value of the &lt;attribute&gt; attribute from the dictionary, represented as a string. When the dictionary does not contain the &lt;attribute&gt;, an empty string is returned. When &lt;index&gt; is provided, return the &lt;index&gt;th instance of the attribute. Some attributes can appear more than once in the request (or response) packet. The &lt;index&gt; argument is used to select which instance to return. When bMore is provided, the get method sets bMore to 1 when more attributes exist after the one returned, and to 0 otherwise. You can use this to determine whether another call to get should be made to retrieve other instances of the attribute.</td>
</tr>
<tr>
<td>isEmpty</td>
<td>$dict isEmpty</td>
<td>Returns 1 when the dictionary has no entries, otherwise returns 0.</td>
</tr>
<tr>
<td>log</td>
<td>$dict log &lt;level&gt; &lt;message&gt; …</td>
<td>Outputs a message into the RADIUS server’s logging system. The &lt;level&gt; should be either LOG_ERROR, LOG_WARNING, or LOG_INFO. The remaining arguments are concatenated together and sent to the logging system at the specified level.</td>
</tr>
</tbody>
</table>
### Table A-1  Tcl Attribute Dictionary Methods (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nextKey</td>
<td>$dict nextKey</td>
<td>Returns the name of the next attribute in the dictionary that follows the attribute returned in the last call to firstKey or nextKey.</td>
</tr>
<tr>
<td>put</td>
<td>$dict put &lt;attribute&gt; &lt;value&gt; [&lt;index&gt;]</td>
<td>Associates &lt;value&gt; with the attribute &lt;attribute&gt; in the dictionary. When &lt;index&gt; is not provided or when &lt;index&gt; equals the special value REPLACE, any existing instances of &lt;attribute&gt; are replaced with the single value. When &lt;index&gt; is provided and equals the special value APPEND, a new instance of &lt;attribute&gt; is appended to the end of the list of instances of the &lt;attribute&gt;. When &lt;index&gt; is provided and is a number, a new instance of &lt;attribute&gt; is inserted at the position indicated. When &lt;index&gt; is provided and equals the special value AUGMENT, only put the attribute when it does not already exist.</td>
</tr>
<tr>
<td>remove</td>
<td>$dict remove &lt;attribute&gt; [&lt;index&gt;]</td>
<td>Removes the &lt;attribute&gt; attribute from the dictionary. When &lt;index&gt; is not provided or when &lt;index&gt; equals the special value REMOVE_ALL, remove any existing instances of &lt;attribute&gt;. When &lt;index&gt; is provided and is a number, remove the instance of &lt;attribute&gt; at the position indicated. Always returns 1, even when the dictionary did not contain the &lt;attribute&gt; at that &lt;index&gt;.</td>
</tr>
<tr>
<td>size</td>
<td>$dict size</td>
<td>Returns the number of entries in the dictionary.</td>
</tr>
<tr>
<td>trace</td>
<td>$dict trace &lt;level&gt; &lt;message&gt; ...</td>
<td>Outputs a message into the packet tracing system used by the RADIUS server. At level 0, no tracing occurs. At level 1, only an indication the server received the packet and sent a reply is output. As the number gets higher, the amount of information output increases, until at level 4, where everything is traced as output. The remaining arguments are concatenated and sent to the tracing system at the specified level.</td>
</tr>
</tbody>
</table>
Tcl Environment Dictionary

A dictionary is a data structure that contains key/value pairs. An Environment dictionary is a dictionary in which the keys and values are constrained to be strings. The Tcl Environment dictionary is used to communicate information from the script to the server and from script to script within the processing of a particular request. Note, there can be only one instance of a key in the Environment dictionary.

Table A-2 lists all the methods you can use with the Request and Response dictionaries.

Table A-2  Tcl Environment Dictionary Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clear</td>
<td>$dict clear</td>
<td>Removes all entries from the dictionary.</td>
</tr>
<tr>
<td>containsKey</td>
<td>$dict containsKey &lt;key&gt;</td>
<td>Returns 1 when the dictionary contains the &lt;key&gt; key, otherwise returns 0.</td>
</tr>
<tr>
<td>firstKey</td>
<td>$dict firstKey</td>
<td>Returns the name of the first key in the dictionary. Note, the keys are not stored sorted by name.</td>
</tr>
<tr>
<td>get</td>
<td>$dict get &lt;key&gt;</td>
<td>Returns the value of &lt;key&gt; from the dictionary. When the dictionary does not contain the &lt;key&gt;, an empty string is returned.</td>
</tr>
<tr>
<td>isEmpty</td>
<td>$dict isEmpty</td>
<td>Returns 1 when the dictionary has no entries, otherwise returns 0.</td>
</tr>
<tr>
<td>log</td>
<td>$dict log &lt;level&gt; &lt;message&gt; …</td>
<td>Outputs a message into the logging system used by the RADIUS server. &lt;level&gt; should be one of LOG_ERROR, LOG_WARNING, or LOG_INFO. The remaining arguments are concatenated together and sent to the logging system at the specified level.</td>
</tr>
<tr>
<td>nextKey</td>
<td>$dict nextKey</td>
<td>Returns the name of the next key in the dictionary that follows the key returned in the last call to firstKey or nextKey.</td>
</tr>
<tr>
<td>put</td>
<td>$dict put &lt;key&gt; &lt;value&gt;</td>
<td>Associates &lt;value&gt; with the &lt;key&gt; key in the dictionary, replacing an existing instance of &lt;key&gt; with the new value.</td>
</tr>
<tr>
<td>remove</td>
<td>$dict remove &lt;key&gt;</td>
<td>Removes the &lt;key&gt; key from the dictionary. Always returns 1, even when the dictionary did not contain the &lt;key&gt;.</td>
</tr>
<tr>
<td>size</td>
<td>$dict size</td>
<td>Returns the number of entries in the dictionary.</td>
</tr>
<tr>
<td>trace</td>
<td>$dict &lt;level&gt; &lt;message&gt; …</td>
<td>Outputs a message into the packet tracing system used by the RADIUS server. At level 0, no tracing occurs. At level 1, only an indication the server received the packet and sent a reply is output. As the number gets higher, the amount of information output is greater, until at level 4, where everything the is traced as output. The remaining arguments are concatenated together and sent to the tracing system at the specified level.</td>
</tr>
</tbody>
</table>
REX Attribute Dictionary

A dictionary is a data structure that contains key/value pairs. An Attribute dictionary is a dictionary in which the keys are constrained to be the attributes as defined in the RADIUS server configuration and the values are constrained to be legal values for that particular attribute. Attribute dictionaries have the unusual feature that there can be more than one instance of a particular key in the dictionary. These instances are ordered, with the first instance at index 0. Some of the methods of an Attribute dictionary allow an index to be specified to indicate a particular instance or position in the list of instances to be referenced.

When writing REX scripts, you can specify keys as the string representation of the name of the attribute or by type, which is a byte sequence defining the attribute. The values can also be specified as the string representation of the value or as the byte sequence, which is the attribute. These options mean some of these access methods have four different variations that are the combinations of string or type for the key, and string or bytes for the value. This section contains the following topics:

- Attribute Dictionary Methods
- REX Environment Dictionary

Attribute Dictionary Methods

Attribute dictionaries use active commands, called methods, that allow you to change and access the values in the dictionaries.

Table A-3 lists all of the methods you can use with the Request and Response dictionaries.

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
</table>
| addProfile       | addProfile(pDict, pszProfile, iMode)                                   | Copies all of the attributes in the <pszProfile> profile into the dictionary. Note, <pszProfile> must be the name of one of the profiles listed in the server configuration. When <iMode> equals the special value REX_REPLACE, it replaces any duplicate instances of the attributes in the dictionary with the attribute from the profile. When <iMode> equals the special value REX_APPEND, it appends a new instance of the attributes to any attributes already in the dictionary. When <iMode> equals the special value.
|                  |                                                                        | When the mode is REX_AUGMENT, it adds the attribute in the dictionary, if it does not already exist in the dictionary. |
| allocateMemory   | allocateMemory(pDict, iSize)                                           | Allocates memory for use in scripts that persist only for the lifetime of this request. This memory is released when processing for this request is complete. |
### REX Attribute Dictionary Methods (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clear</td>
<td><code>pDict-&gt;clear(rex_AttributeDictionary_t* pDict)</code></td>
<td>Removes all entries from the dictionary.</td>
</tr>
<tr>
<td>containsKey</td>
<td><code>pDict-&gt;containsKey(rex_AttributeDictionary_t* pDict, const char* &lt;pszAttribute&gt;)</code></td>
<td>Returns TRUE when the dictionary contains <code>&lt;pszAttribute&gt;</code>, otherwise returns FALSE.</td>
</tr>
<tr>
<td>containsKeyByType</td>
<td><code>pDict-&gt;containsKeyByType(rex_AttributeDictionary_t* pDict, const bytes_t* &lt;pAttribute&gt;)</code></td>
<td>Returns TRUE when the dictionary contains <code>&lt;pAttribute&gt;</code>, otherwise returns FALSE.</td>
</tr>
<tr>
<td>firstKey</td>
<td><code>pDict-&gt;firstKey(rex_AttributeDictionary_t* pDict)</code></td>
<td>Returns the name of the first attribute in the dictionary. Note, the attributes are not stored in a sorted order of name.</td>
</tr>
<tr>
<td>firstKeyByType</td>
<td><code>pDict-&gt;firstKeyByType(rex_AttributeDictionary_t* pDict)</code></td>
<td>Returns a pointer to the byte sequence defining the first attribute in the dictionary. Note, attributes are not stored sorted by name.</td>
</tr>
<tr>
<td>get</td>
<td><code>pDict-&gt;get(rex_AttributeDictionary_t* pDict, const char* pszAttribute, int &lt;iIndex&gt;, abool_t* &lt;pbMore&gt;)</code></td>
<td>Returns the value of the <code>&lt;iIndex&gt;'d instance of the attribute from the dictionary, represented as a string. When the dictionary does not contain the attribute (or that many instances of the attribute), an empty string is returned. When </code>&lt;pbMore&gt;<code>is non-zero, the get method sets</code>&lt;pbMore&gt;` to TRUE when more instances of the attribute exist after the one returned, and to FALSE otherwise. This can be used to determine whether another call to get should be made to retrieve other instances of the attribute.</td>
</tr>
</tbody>
</table>
### Table A-3: REX Attribute Dictionary Methods (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>getBytes</td>
<td><code>const abytes_t* pDict-&gt;getBytes(rex_AttributeDictionary_t* pDict, const char* pszAttribute, int &lt;iIndex&gt;, abool_t* &lt;pbMore&gt;)</code></td>
<td>Returns the value of the <code>&lt;Index&gt;</code>'d instance of the attribute from the dictionary, as a sequence of bytes. When the dictionary does not contain the attribute (or that many instances of the attribute), 0 is returned. When <code>&lt;pbMore&gt;</code> is non-zero, the <code>getBytes</code> method sets <code>&lt;pbMore&gt;</code> to TRUE when more instances of the attribute exist after the one returned, and to FALSE otherwise. This can be used to determine whether another call to <code>getBytes</code> should be made to retrieve other instances of the attribute.</td>
</tr>
<tr>
<td>getBytesByType</td>
<td><code>const abytes_t* pDict-&gt;getBytesByType (rex_AttributeDictionary_t* pDict, const abytes_t* pAttribute, int &lt;iIndex&gt;, abool_t* &lt;pbMore&gt;)</code></td>
<td>Returns the value of the <code>&lt;Index&gt;</code>'d instance of the attribute from the dictionary, as a sequence of bytes. When the dictionary does not contain the attribute (or that many instances of the attribute), 0 is returned instead. When <code>&lt;pbMore&gt;</code> is non-zero, sets the variable pointed to TRUE when more instances of the attribute exist after the one returned, and to FALSE otherwise. This can be used to determine whether another call to <code>getBytesByType</code> should be made to retrieve other instances of the attribute.</td>
</tr>
<tr>
<td>getByType</td>
<td><code>const char* pDict-&gt;get(rex_AttributeDictionary_t* pDict, const abytes_t* &lt;pAttribute&gt;, int &lt;iIndex&gt;, abool_t* &lt;pbMore&gt;)</code></td>
<td>Returns the value of the <code>&lt;Index&gt;</code>'d instance of the attribute from the dictionary, as represented as a string. When the dictionary does not contain the attribute (or that many instances of the attribute), returns an empty string. When <code>&lt;pbMore&gt;</code> is non-zero, the <code>getByType</code> method sets <code>&lt;pbMore&gt;</code> to TRUE when more instances of the attribute exist after the one returned, and to FALSE otherwise. This can be used to determine whether another call to <code>getByType</code> should be made to retrieve other instances of the attribute.</td>
</tr>
<tr>
<td>getType</td>
<td><code>const char* pDict-&gt;getType(rex_AttributeDictionary_t* pDict, const abytes_t* &lt;pAttribute&gt;)</code></td>
<td>Returns a pointer to the byte sequence defining the attribute, when the attribute name matches a configured attribute, zero otherwise.</td>
</tr>
</tbody>
</table>
### Table A-3 REX Attribute Dictionary Methods (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>isEmpty</td>
<td>abool_t pDict-&gt;isEmpty(rex_AttributeDictionary_t* pDict)</td>
<td>Returns TRUE when the dictionary has 0 entries, FALSE otherwise.</td>
</tr>
</tbody>
</table>
| log        | abool_t pDict->log(rex_AttributeDictionary_t* pDict, int <iLevel>, const char* <pszFormat>, ...) | Outputs a message into the logging system used by the RADIUS server.  
<iLevel> should be one of REX_LOG_ERROR, REX_LOG_WARNING, or REX_LOG_INFO. The pszFormat argument is treated as a printf-style format string, and it, along with the remaining arguments, are formatted and sent to the logging system at the specified level. |
| nextKey    | const char* pDict->nextKey(rex_AttributeDictionary_t* pDict)           | Returns the name of the next attribute in the dictionary that follows the attribute returned in the last call to firstKey or nextKey. |
| nextKeyByType | const abyttes_t* pDict->nextKeyByType(rex_AttributeDictionary_t* pDict) | Returns a pointer to the byte sequence defining the next attribute in the dictionary that follows the attribute returned in the last call to firstKeyByType or nextKeyByType. |
| put        | abool_t pDict->put(rex_AttributeDictionary_t* pDict, const char* <pszAttribute>, const char* <pszValue>, int <iIndex>) | Converts <pszValue> to a sequence of bytes, according to the definition of <pszAttribute> in the server configuration. Associates that sequence of bytes with <pszAttribute> in the dictionary. When <iIndex> equals the special value REX_REPLACE, it replaces any existing instances of <pszAttribute> with a single value. When <iIndex> equals the special value REX_APPEND, it appends a new instance of <pszAttribute> to the end of the list of existing instances of <pszAttribute>. Otherwise, a new instance of <pszAttribute> is inserted at the position indicated. This method returns TRUE unless <pszAttribute> does not match any configured attributes or the value could not be converted to a legal value. When <iIndex> equals the special value REX_AUGMENT, only put <pszAttribute> when it does not already exist. |
### Table A-3 REX Attribute Dictionary Methods (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>putBytes</td>
<td>abool_t pDict-&gt;putBytes(rex_AttributeDictionary_t* pDict, const char* pszAttribute, const abytes_t* pValue, int iIndex)</td>
<td>Associates (&lt;pValue&gt;) with the attribute (&lt;pszAttribute&gt;) in the dictionary. When (&lt;iIndex&gt;) equals the special value REX_REPLACE, it replaces any existing instances of the (&lt;pszAttribute&gt;) with a single new value. When (&lt;iIndex&gt;) equals the special value REX_APPEND, it appends a new instance of (&lt;pszAttribute&gt;) to the end of the list of existing instances of (&lt;pszAttribute&gt;). When (&lt;iIndex&gt;) equals the special value REX_AUGMENT, only put the (&lt;pszAttribute&gt;) when it does not already exist. Otherwise, a new instance of (&lt;pszAttribute&gt;) is inserted at the position indicated. This method returns TRUE unless the attribute name does not match any configured attributes.</td>
</tr>
<tr>
<td>putBytesByType</td>
<td>abool_t pDict-&gt;putBytesByType(rex_AttributeDictionary_t* pDict, const abytes_t* pAttribute, const abytes_t* pValue, int iIndex)</td>
<td>Associates (&lt;pValue&gt;) with the attribute (&lt;pAttribute&gt;) in the dictionary. When (&lt;iIndex&gt;) equals the special value REX_REPLACE, it replaces any existing instances of (&lt;pAttribute&gt;) with the new value. When (&lt;iIndex&gt;) equals the special value REX_APPEND, it appends a new instance of (&lt;pAttribute&gt;) to the end of the list of existing instances of (&lt;pAttribute&gt;). When (&lt;iIndex&gt;) equals the special value REX_AUGMENT, only put (&lt;pAttribute&gt;) when it does not already exist. Otherwise, insert a new instance of (&lt;pAttribute&gt;) at the position indicated. This method returns TRUE unless the attribute name does not match any configured attributes.</td>
</tr>
</tbody>
</table>
### Table A-3  REX Attribute Dictionary Methods (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>putByType</td>
<td>abool_t pDict-&gt;putByType(rex_AttributeDictionary_t* pDict, const abbytes_t* &lt;pszAttribute&gt;, const char* &lt;pszValue&gt;, int &lt;iIndex&gt;)</td>
<td>Converts &lt;pszValue&gt; to a sequence of bytes, according to the definition of &lt;pszAttribute&gt; in the server configuration. Associates that sequence of bytes with &lt;pszAttribute&gt; in the dictionary. When &lt;iIndex&gt; equals the special value REX_REPLACE, it replaces any existing instances of &lt;pszAttribute&gt; with a single new value. When &lt;iIndex&gt; equals the special value REX_APPEND, it appends a new instance of &lt;pszAttribute&gt; to the end of the list of existing instances of &lt;pszAttribute&gt;. Otherwise, it inserts a new instance of &lt;pszAttribute&gt; at the position indicated. This method returns TRUE unless &lt;pszAttribute&gt; does not match any configured attributes, or the value could not be converted to a legal value.</td>
</tr>
<tr>
<td>remove</td>
<td>abool_t pDict-&gt;remove(rex_AttributeDictionary_t* pDict, const char* &lt;pszAttribute&gt;, int &lt;iIndex&gt;)</td>
<td>Removes the &lt;pszAttribute&gt; from the dictionary. When &lt;iIndex&gt; equals the special value REX_REMOVE_ALL, removes any existing instances of &lt;pszAttribute&gt;. Otherwise, it removes the instance of &lt;pszAttribute&gt; at the position indicated. Returns TRUE, even when the dictionary did not contain &lt;pszAttribute&gt; at the &lt;iIndex&gt;, unless &lt;pszAttribute&gt; does not match any configured attribute.</td>
</tr>
<tr>
<td>removeByType</td>
<td>abool_t pDict-&gt;removeByType(rex_AttributeDictionary_t* pDict, const abbytes_t* &lt;pAttribute&gt;, int &lt;iIndex&gt;)</td>
<td>Removes the &lt;pAttribute&gt; from the dictionary. When &lt;iIndex&gt; equals the special value REX_REMOVE_ALL, it removes any existing instances of &lt;pszAttribute&gt;. Otherwise, the instance of &lt;pAttribute&gt; at the position indicated is removed. Always returns TRUE, even when the dictionary did not contain &lt;pAttribute&gt; at the &lt;iIndex&gt;.</td>
</tr>
<tr>
<td>reschedule</td>
<td>abool_t pDict-&gt;reschedule(rex_AttributeDictionary_t* pDict)</td>
<td>Enables control over asynchronous activities. It enables you to collect similar activities and mark them as pending. You can then process them and reschedule them. You can only use this attribute with multithreaded services. Use caution when employing this method.</td>
</tr>
</tbody>
</table>
REX Environment Dictionary

A dictionary is a data structure that contains key/value pairs. An Environment dictionary is a dictionary in which the keys and values are constrained to be strings. The REX Environment dictionary is used to communicate information from the script to the server and from script to script within the processing of a particular request. Note, there can be only one instance of a key in the Environment dictionary.

REX Environment Dictionary Methods

The Environment dictionary uses active commands, called methods, to allow you to change and access the values in the dictionary. Table A-4 lists all of the methods you can use with the REX Environment dictionary.

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>allocateMemory</td>
<td>void* pDict-&gt;allocateMemory(rex_EnvironmentDictionary_t* pDict, unsigned int &lt;iSize&gt;)</td>
<td>Allocate memory for use in scripts that persist only for the lifetime of this request. This memory is released when processing for this request is complete.</td>
</tr>
<tr>
<td>clear</td>
<td>void pDict-&gt;clear(rex_EnvironmentDictionary_t* pDict)</td>
<td>Removes all entries from the dictionary.</td>
</tr>
<tr>
<td>containsKey</td>
<td>abool_t pDict-&gt;containsKey(rex_EnvironmentDictionary_t* pDict, const char* &lt;pszKey&gt;)</td>
<td>Returns TRUE when the dictionary contains &lt;pszKey&gt;, otherwise returns FALSE.</td>
</tr>
<tr>
<td>firstKey</td>
<td>const char* pDict-&gt;firstKey(rex_EnvironmentDictionary_t* pDict)</td>
<td>Returns the name of the first key in the dictionary. Note, the keys are not stored sorted by name.</td>
</tr>
</tbody>
</table>
### Table A-4  REX Environment Dictionary Methods (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td><code>const char* pDict-&gt;get(rex_EnvironmentDictionary_t* pDict, const char* &lt;pszKey&gt;)</code></td>
<td>Returns the value associated with <code>&lt;pszKey&gt;</code> from the dictionary. When the dictionary does not contain <code>&lt;pszKey&gt;</code>, an empty string is returned.</td>
</tr>
<tr>
<td>isEmpty</td>
<td><code>abool_t pDict-&gt;isEmpty(rex_EnvironmentDictionary_t* pDict)</code></td>
<td>Returns TRUE when the dictionary has 0 entries, FALSE otherwise.</td>
</tr>
<tr>
<td>log</td>
<td><code>abool_t pDict-&gt;log(rex_EnvironmentDictionary_t* pDict, int &lt;iLevel&gt;, const char* &lt;pszFormat&gt;, ...)</code></td>
<td>Outputs a message into the logging system used by the RADIUS server. <code>&lt;iLevel&gt;</code> should be one of <code>REX_LOG_ERROR</code>, <code>REX_LOG_WARNING</code>, or <code>REX_LOG_INFO</code>. The <code>&lt;pszFormat&gt;</code> argument is treated as a <code>printf</code>-style format string, and it, along with the remaining arguments, are formatted and sent to the logging system at the specified level.</td>
</tr>
<tr>
<td>nextKey</td>
<td><code>const char* pDict-&gt;nextKey(rex_EnvironmentDictionary_t* pDict)</code></td>
<td>Returns the name of the next key in the dictionary that follows the key returned in the last call to <code>firstKey</code> or <code>nextKey</code>.</td>
</tr>
<tr>
<td>put</td>
<td><code>abool_t pDict-&gt;put(rex_EnvironmentDictionary_t* pDict, const char* &lt;pszValue&gt;, const char* &lt;pszKey&gt;)</code></td>
<td>Associates the value with <code>&lt;pszKey&gt;</code> in the dictionary, replacing any existing instance of <code>&lt;pszKey&gt;</code> with the new <code>&lt;pszValue&gt;</code>.</td>
</tr>
<tr>
<td>remove</td>
<td><code>abool_t pDict-&gt;remove(rex_EnvironmentDictionary_t* pDict, const char* &lt;pszKey&gt;)</code></td>
<td>Removes <code>&lt;pszKey&gt;</code> and the associated value from the dictionary. Always returns TRUE, even when the dictionary did not contain <code>&lt;pszKey&gt;</code>.</td>
</tr>
<tr>
<td>reschedule</td>
<td><code>abool_t pDict-&gt;reschedule(rex_AttributeDictionary_t* pDict)</code></td>
<td>Enables control over asynchronous activities. It enables you to collect similar activities and mark them as pending. You can then process them and reschedule them. You can only use this attribute with multithreaded services. Use caution when employing this method.</td>
</tr>
</tbody>
</table>
Java Attribute Dictionary

The AttributeDictionary is a dictionary of attributes, where the keys are the attribute types and the values are the data fields in the attribute. Both keys and values must conform to the definition of attributes in the server's Attribute Dictionary. Keys (types) can be either strings or byte arrays. If strings, they are the names of attributes. If byte arrays, they are the binary type. The type associated with a name can be retrieved by calling the static method getType(java.lang.String). Using byte arrays is slightly more efficient - methods that take String keys must do the mapping from String to byte array in the course of executing the method. Similarly, values can be strings or byte arrays. Again, string values are converted to the appropriate binary representation when stored in an AttributeDictionary and back again when retrieved into a string variable.

Keys in an AttributeDictionary can be associated with multiple values. Each of the values associated with a key is ordered with an integer index denoting its position in the list of values. Given an AttributeDictionary, a key and an index, each value associated with a key can be looked up. This section contains the following topics:

- Java Environment Dictionary Methods
- Interface Extension Methods
- Interface ExtensionforSession Methods
- Interface ExtensionwithInitialization Methods
- Interface ExtensionforSessionwithInitialization Methods
- Variables in the Marker Extension Interface
- Session Record Methods

Java Attribute Dictionary Methods

Attribute dictionaries use active commands called methods, that allow you to change and access the values in the dictionaries.
Table A-5 lists all of the methods you can use with the Request and Response dictionaries.

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>size</td>
<td>public int size()</td>
<td>Returns the number of distinct keys in the dictionary.</td>
</tr>
<tr>
<td>isEmpty</td>
<td>public boolean isEmpty()</td>
<td>Tests if the dictionary contains any entries.</td>
</tr>
<tr>
<td>clear</td>
<td>public void clear()</td>
<td>Removes all entries from the dictionary.</td>
</tr>
<tr>
<td>containsKey</td>
<td>public boolean containsKey(java.lang.String key)</td>
<td>Returns true if an entry exists for key.</td>
</tr>
<tr>
<td>get</td>
<td>public java.lang.String get(java.lang.String key)</td>
<td>Returns the first value associated with the key.</td>
</tr>
<tr>
<td>get</td>
<td>public java.lang.String get(java.lang.String key, int index)</td>
<td>Returns the value at position index associated with the key.</td>
</tr>
<tr>
<td>put</td>
<td>public boolean put(java.lang.String key, java.lang.String value)</td>
<td>Associates key with a value. Any existing values associated with the key are removed before adding this association.</td>
</tr>
<tr>
<td>put</td>
<td>public boolean put(java.lang.String key, java.lang.String value, int index)</td>
<td>Associates key with a value depending on the value of index. If index equals Extension.EXT_REPLACE, any existing values are removed before adding this new association. If index equals Extension.EXT_APPEND, a new value is added at the end of the list of existing values. If index equals Extension.EXT_AUGMENT, the new association is only made if the dictionary does not already have an entry for key. If index is a number greater than or equal to 0 and less than the number of entries in the list, the value is inserted at that position in the list. Otherwise, the value is appended at the end of the list.</td>
</tr>
<tr>
<td>getBytes</td>
<td>public byte[] getBytes(java.lang.String key)</td>
<td>Returns the first value associated with the key.</td>
</tr>
<tr>
<td>getBytes</td>
<td>public byte[] getBytes(java.lang.String key, int index)</td>
<td>Returns the value at position index associated with key.</td>
</tr>
<tr>
<td>putBytes</td>
<td>public boolean putBytes(java.lang.String key, byte[] value)</td>
<td>Associates key with value. Any existing values associated with key are removed before adding this association.</td>
</tr>
<tr>
<td>Name</td>
<td>Syntax</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>putBytes</td>
<td>public boolean putBytes(java.lang.String key, byte[] value, int index)</td>
<td>Associates key with a value depending on the value of index. If index equals Extension.EXT_REPLACE, any existing values are removed before adding this new association. If index equals Extension.EXT_APPEND, a new value is added at the end of the list of existing values. If index equals Extension.EXT_AUGMENT, the new association is only made if the dictionary does not already have an entry for key. If index is a number greater than or equal to 0 and less than the number of entries in the list, the value is inserted at that position in the list. Otherwise, the value is appended at the end of the list.</td>
</tr>
<tr>
<td>remove</td>
<td>public void remove(java.lang.String key)</td>
<td>Removes key (and all corresponding values) from the dictionary. This method does nothing if key is not in the dictionary.</td>
</tr>
<tr>
<td>remove</td>
<td>public void remove(java.lang.String key, int index)</td>
<td>Removes value at the position index that is associated with key. If the index equals Extension.EXT_REMOVE_ALL or if the value being removed is the last value associated with key, the key is removed from the dictionary. This method does nothing if key is not in the dictionary.</td>
</tr>
<tr>
<td>addProfile</td>
<td>public boolean addProfile(java.lang.String profileName)</td>
<td>Adds all the attributes contained in the specified profile into the dictionary. Any existing attributes that have the same keys as attributes in the profile are removed before adding the new attributes.</td>
</tr>
<tr>
<td>AddProfile</td>
<td>boolean addProfile(java.lang.String profileName, int mode)</td>
<td>Adds all the attributes contained in the specified profile into the dictionary. Any existing attributes that have the same keys as attributes in the profile will be treated depending on the mode value. For each attribute in the Profile, if mode equals Extension.EXT_REPLACE, any values associated with the attribute in the dictionary are removed before adding the attribute. If index equals Extension.EXT_APPEND, a new value is added at the end of the list of existing values. If index equals Extension.EXT_AUGMENT, a new value is added only if the dictionary does not already have an entry for the given key.</td>
</tr>
<tr>
<td>getType</td>
<td>public static byte[] getType(java.lang.String key)</td>
<td>Takes the name of the attribute (as a string) and returns the binary form of key.</td>
</tr>
</tbody>
</table>
Java Environment Dictionary

The Environment Dictionary can be used to store information between Extensions invoked subsequently on a given request or can be used to pass information between the Extension and the server properly.

The Environment Dictionary maps keys to values, where the keys and values are strings. In any one instance of the Environment Dictionary, every key is associated with at most one value. Given an Environment Dictionary and a key, the associated value can be looked up. Any non-null string can be used as a key and value.

Java Environment Dictionary Methods

The Environment dictionary uses active commands called methods, to allow you to change and access the values in the dictionary. Table A-6 lists all of the methods you can use with the java Environment dictionary.

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>size</td>
<td>public int size()</td>
<td>Returns the number of entries (distinct keys) in the dictionary.</td>
</tr>
<tr>
<td>isEmpty</td>
<td>public boolean isEmpty()</td>
<td>Tests if the dictionary contains no entries.</td>
</tr>
<tr>
<td>clear</td>
<td>public void clear()</td>
<td>Removes all entries from the dictionary.</td>
</tr>
<tr>
<td>containsKey</td>
<td>public boolean containsKey(java.lang.String key)</td>
<td>Returns true if the dictionary contains an entry for key.</td>
</tr>
<tr>
<td>get</td>
<td>public java.lang.String get(java.lang.String key)</td>
<td>Returns the value associated with key in the dictionary.</td>
</tr>
</tbody>
</table>
### Interface Extension

Classes that are going to be used as scripts or services from Access Registrar must implement the Extension interface. When a Java scripting point or service is encountered during the processing of a request, the server will call the runExtension method defined in this interface and implemented by the appropriate class.

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>remove</td>
<td>public void remove(java.lang.String key)</td>
<td>Removes key (and its corresponding value) from this dictionary. This method does nothing if key is not in the dictionary.</td>
</tr>
<tr>
<td>keys</td>
<td>public java.util Enumeration keys()</td>
<td>Returns an enumeration of the keys in the dictionary.</td>
</tr>
<tr>
<td>elements</td>
<td>public java.util Enumeration elements()</td>
<td>Returns an enumeration of the entries in the dictionary.</td>
</tr>
<tr>
<td>log</td>
<td>public static void log(int level, java.lang.String message)</td>
<td>Prints a message in the server log at the specified level.</td>
</tr>
<tr>
<td>trace</td>
<td>public void trace(int level, java.lang.String message)</td>
<td>Prints a message in the server trace file at the specified level.</td>
</tr>
<tr>
<td>reschedule</td>
<td>public void reschedule()</td>
<td>Informs the server that it should take back ownership of the request associated with the dictionary and continue processing it.</td>
</tr>
</tbody>
</table>
Interface Extension Methods

Table A-7 lists the methods you can use for interface extension

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>runExtension</td>
<td>int runExtension(int iExtensionPoint, AttributeDictionary request, AttributeDictionary response, EnvironmentDictionary environment)</td>
<td>This method is called whenever a Java scripting point or service is encountered during the processing of a request.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>When runExtension is used as a script, it should process requests as quickly as possible, without blocking. This is because the server has a limited number of threads that it is using to process requests and if any one extension takes too long to run, it is likely that many requests will be delayed as each one calls the extension. runExtension must return either EXT_OK to indicate that processing of this request should continue or EXT_ERROR to indicate that an error occurred while processing this request and that the request should be dropped. Extensions should always log an error before returning EXT_ERROR so that the administrator has a way to determine the problem that was encountered. When runExtension is used as a service, it will be called once before requests start coming in (with the iExtensionPoint parameter set to EXT_START_SERVICE) to give the extension the opportunity to initialize resources needed to process requests, and once after the last request has been received (with the iExtensionPoint parameter set to EXT_STOP_SERVICE) to give the extension the opportunity to release those resources before stopping. runExtension must return one of the following values: EXT_OK, EXT_ERROR or EXT_PENDING. EXT_PENDING should be returned to inform the server that the extension has taken ownership of the request, will process the request on a background thread, and will inform the server when it is time to continue processing the request by calling reschedule() on one of the request’s dictionaries.</td>
</tr>
</tbody>
</table>

Interface Extension for Session

Classes that are going to be used as scripts at Session Manager level from Cisco Prime Access Registrar must implement the ExtensionForSession interface. When a Java scripting point or service is encountered during the processing of a request, the server will call the runExtension method defined in this interface and implemented by the appropriate class.
Interface Extension for session Methods

Table A-8 lists the methods you can use for interface extension for session

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>runExtension</td>
<td>int runExtension(int iExtensionPoint, AttributeDictionary request, AttributeDictionary response, EnvironmentDictionary environment, SessionRecord session)</td>
<td>This method is called whenever a Java scripting point or service is encountered during the processing of a request. When runExtension is used as a script, it should process requests as quickly as possible, without blocking. This is because the server has a limited number of threads that it is using to process requests and if any one extension takes too long to run, it is likely that many requests will be delayed as each one calls the extension. runExtension must return either EXT_OK to indicate that processing of this request should continue or EXT_ERROR to indicate that an error occurred while processing this request and that the request should be dropped. Extensions should always log an error before returning EXT_ERROR so that the administrator has a way to determine the problem that was encountered. When runExtension is used as a service, it will be called once before requests start coming in (with the iExtensionPoint parameter set to EXT_START_SERVICE) to give the extension the opportunity to initialize resources needed to process requests, and once after the last request has been received (with the iExtensionPoint parameter set to EXT_STOP_SERVICE) to give the extension the opportunity to release those resources before stopping. runExtension must return one of the following values: EXT_OK, EXT_ERROR or EXT_PENDING. EXT_PENDING should be returned to inform the server that the extension has taken ownership of the request, will process the request on a background thread, and will inform the server when it is time to continue processing the request by calling reschedule() on one of the request’s dictionaries.</td>
</tr>
</tbody>
</table>

Interface Extension with initialization

Classes that are going to be used as scripts or services from Access Registrar implements the ExtensionWithInitialization interface. ExtensionWithInitialization extends the Extension interface with methods to initialize and destroy the extension. initialize(java.lang.String) is called when the extension is first loaded, with the string argument being set from the InitializeArg property that was defined in the server configuration when the extension was defined (either as a Script or a Service). Destroy() is called before the extension is unloaded.
Interface Extension with Initialization Methods

Table A-9 lists the methods you can use for Interface Extension with Initialization.

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>initialize</td>
<td>void initialize(java.lang.String initializeArg)</td>
<td>This method is called by the server when the extension is first loaded.</td>
</tr>
<tr>
<td>destroy</td>
<td>void destroy()</td>
<td>This method is called by the server when the extension is going to be unloaded.</td>
</tr>
</tbody>
</table>

Interface Extension for Session with Initialization

Classes that are going to be used as scripts from Access Registrar at Session Manager level implement the ExtensionForSessionWithInitialization interface. ExtensionForSessionWithInitialization extends the ExtensionForSession interface with methods to initialize and destroy the extension.

initialize(java.lang.String) is called when the extension is first loaded, with the string argument being set from the InitializeArg property that was defined in the server configuration when the extension was defined (either as a script or a service). Destroy () is called before the extension is unloaded.

Interface Extension for Session with Initialization Methods

Table A-10 lists the methods you can use for Interface Extension for Session with Initialization.

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>initialize</td>
<td>void initialize(java.lang.String initializeArg)</td>
<td>This method is called by the server when the extension is first loaded.</td>
</tr>
<tr>
<td>destroy</td>
<td>void destroy()</td>
<td>This method is called by the server when the extension is going to be unloaded.</td>
</tr>
</tbody>
</table>

Interface Marker Extension

This is just going to be a marker interface containing various member variables which can be used in interfaces/classes extending from this interface. Extension and ExtensionForSession interfaces will extend this interface.
## Variables in the Marker Extension Interface

Table A-11 lists the variables in the marker extension interface.

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXT_OK</td>
<td>static final int EXT_OK</td>
<td>Returns EXT_OK by implementation of runExtension() to indicate that the extension operated correctly and processing of the request should continue.</td>
</tr>
<tr>
<td>EXT_ERROR</td>
<td>static final int EXT_ERROR</td>
<td>Returns EXT_ERROR by implementation of runExtension() to indicate that the extension failed in some way and processing of the request should NOT continue.</td>
</tr>
<tr>
<td>EXT_PENDING</td>
<td>static final int EXT_PENDING</td>
<td>Returns EXT_PENDING by implementations of runExtension() to indicate that the extension operated correctly and the extension wants to take ownership of the request for a while. Further processing of the request by the server will be postponed until the extension indicates that it can do so by calling the reschedule method on any of the dictionaries.</td>
</tr>
<tr>
<td>EXT_LOG_ERROR</td>
<td>static final int EXT_LOG_ERROR</td>
<td>Indicates that the message should be logged with a severity of ERROR, when passed to log() in the level parameter.</td>
</tr>
<tr>
<td>EXT_LOG_WARNING</td>
<td>static final int EXT_LOG_WARNING</td>
<td>Indicates that the message should be logged with a severity of WARNING, when passed to log() in the level parameter.</td>
</tr>
<tr>
<td>EXT_LOG_INFO</td>
<td>static final int EXT_LOG_INFO</td>
<td>Indicates that the message should be logged with a severity of INFO, when passed to log() in the level parameter.</td>
</tr>
<tr>
<td>EXT_REMOVE_ALL</td>
<td>static final int EXT_REMOVE_ALL</td>
<td>Indicates that all values associated with the specified key should be removed, when passed to AttributeDictionary::remove() in the index parameter.</td>
</tr>
<tr>
<td>EXT_REPLACE</td>
<td>static final int EXT_REPLACE</td>
<td>Indicates that all existing values associated with the specified key(s) should be removed before adding the new value(s), when passed to AttributeDictionary::put() (and its variants) in the index parameter or to AttributeDictionary::addProfile() in the mode parameter.</td>
</tr>
<tr>
<td>EXT_APPEND</td>
<td>static final int EXT_APPEND</td>
<td>Indicates that the new value(s) should be appended to the end of the list of any existing values associated with the specified key(s), when passed to AttributeDictionary::put() (and its variants) in the index parameter or to AttributeDictionary::addProfile() in the mode parameter.</td>
</tr>
</tbody>
</table>
### Java Attribute Dictionary

#### Table A-11  Marker Extension Interface Variables (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXT_AUGMENT</td>
<td>static final int EXT_AUGMENT</td>
<td>Indicates that the new association(s) should only be added if the dictionary does not already have an entry for the given key(s), when passed to AttributeDictionary::put() (and its variants) in the index parameter or to AttributeDictionary::addProfile() in the mode parameter.</td>
</tr>
<tr>
<td>EXT_START_SERVICE</td>
<td>static final int EXT_START_SERVICE</td>
<td>Indicates that the extension should do whatever is necessary to prepare to offer service, when passed to extensions used as services. This may include starting background threads, opening database connections, and so on.</td>
</tr>
<tr>
<td>EXT_AUTHENTICATION_SERVICE</td>
<td>static final int EXT_AUTHENTICATION_SERVICE</td>
<td>Indicates that the extension should authenticate the current request, when passed to extensions used as services. To indicate whether the request was authenticated or not, the extension should set the EnvironmentDictionary entry for &quot;Response-Type&quot; to either &quot;Access-Accept&quot; or &quot;Access-Reject&quot;.</td>
</tr>
<tr>
<td>EXT_AUTHORIZATION_SERVICE</td>
<td>static final int EXT_AUTHORIZATION_SERVICE</td>
<td>Indicates that the extension should authorize the current request, when passed to extensions used as services.</td>
</tr>
<tr>
<td>EXT_AUTHENTICATION_AND_AUTHORIZATION_SERVICE</td>
<td>static final int EXT_AUTHENTICATION_AND_AUTHORIZATION_SERVICE</td>
<td>Indicates that the extension should both authenticate and authorize the current request, when passed to extensions used as services. To indicate whether the request was authenticated or not, the extension should set the EnvironmentDictionary entry for &quot;Response-Type&quot; to either &quot;Access-Accept&quot; or &quot;Access-Reject&quot;.</td>
</tr>
<tr>
<td>EXT_ACCOUNTING_SERVICE</td>
<td>static final int EXT_ACCOUNTING_SERVICE</td>
<td>Indicates that the extension should produce an accounting record for the current request, when passed to extensions used as services.</td>
</tr>
<tr>
<td>EXT_STOP_SERVICE</td>
<td>static final int EXT_STOP_SERVICE</td>
<td>Indicates that the extension should do whatever is necessary to shut down, when passed to extensions used as services. This may include stopping background threads, closing database connections and so on.</td>
</tr>
<tr>
<td>EXT_NAS_STARTED_ACCOUNTING_SERVICE</td>
<td>static final int EXT_NAS_STARTED_ACCOUNTING_SERVICE</td>
<td>Indicates that the NAS identified in the EnvironmentDictionary (by either the &quot;NAS-Identifier&quot; or &quot;NAS-IP-Address&quot; entries) has indicated that it is starting up, when passed to extensions used as services. This may be used by extensions to prepare to receive requests from this particular NAS if the extension treats requests from different NASs differently.</td>
</tr>
</tbody>
</table>
### Table A-11  Marker Extension Interface Variables (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXT_NAS_STOPPED_ACCOUNTING_SERVICE</td>
<td>static final int EXT_STOPPED_ACCOUNTING_SERVICE</td>
<td>Indicates that the NAS identified in the EnvironmentDictionary (by either the &quot;NAS-Identifier&quot; or &quot;NAS-IP-Address&quot; entries) has indicated that it is shutting down, when passed to extensions used as services. This may be used by extensions to recover any resources associated with this NAS if the extension treats requests from different NASs differently.</td>
</tr>
<tr>
<td>EXT_INCOMING_SERVER_SCRIPTING_POINT</td>
<td>static final int EXT_INCOMING_SERVER_SCRIPTING_POINT</td>
<td>Indicates that the extension is being called from the script /Radius/IncomingScript, when passed to extensions used as scripts.</td>
</tr>
<tr>
<td>EXT_INCOMING_VENDOR_SCRIPTING_POINT</td>
<td>static final int EXT_INCOMING_VENDOR_SCRIPTING_POINT</td>
<td>Indicates that the extension is being called from the script /Radius/Vendors/&lt;vendor&gt;/IncomingScript, when passed to extensions used as scripts.</td>
</tr>
<tr>
<td>EXT_INCOMING_CLIENT_SCRIPTING_POINT</td>
<td>static final int EXT_INCOMING_CLIENT_SCRIPTING_POINT</td>
<td>Indicates that the extension is being called from the script /Radius/Clients/&lt;client&gt;/IncomingScript or from the script /Radius/RemoteServers/&lt;server&gt;/IncomingScript, when passed to extensions used as scripts.</td>
</tr>
<tr>
<td>EXT_INCOMING_SERVICE_SCRIPTING_POINT</td>
<td>static final int EXT_INCOMING_SERVICE_SCRIPTING_POINT</td>
<td>Indicates that the extension is being called from the script /Radius/Services/&lt;service&gt;/IncomingScript, when passed to extensions used as scripts.</td>
</tr>
<tr>
<td>EXT_USERGROUP_AUTHENTICATION_SCRIPTING_POINT</td>
<td>static final int EXT_USERGROUP_AUTHENTICATION_SCRIPTING_POINT</td>
<td>Indicates that the extension is being called from the script /Radius/UserGroups/&lt;group&gt;/AuthenticationScript, when passed to extensions used as scripts.</td>
</tr>
<tr>
<td>EXT_USERRECORD_AUTHENTICATION_SCRIPTING_POINT</td>
<td>static final int EXT_USERRECORD_AUTHENTICATION_SCRIPTING_POINT</td>
<td>Indicates that the extension is being called from the script /Radius/UserLists/&lt;userlist&gt;/&lt;user&gt;/AuthenticationScript, when passed to extensions used as scripts.</td>
</tr>
<tr>
<td>EXT_USERGROUP_AUTHORIZATION_SCRIPTING_POINT</td>
<td>static final int EXT_USERGROUP_AUTHORIZATION_SCRIPTING_POINT</td>
<td>Indicates that the extension is being called from the script /Radius/UserGroups/&lt;group&gt;/AuthorizationScript, when passed to extensions used as scripts.</td>
</tr>
<tr>
<td>EXT_USERRECORD_AUTHORIZATION_SCRIPTING_POINT</td>
<td>static final int EXT_USERRECORD_AUTHORIZATION_SCRIPTING_POINT</td>
<td>Indicates that the extension is being called from the script /Radius/UserLists/&lt;userlist&gt;/&lt;user&gt;/AuthorizationScript, when passed to extensions used as scripts.</td>
</tr>
<tr>
<td>EXT_OUTGOING_SERVICE_SCRIPTING_POINT</td>
<td>static final int EXT_OUTGOING_SERVICE_SCRIPTING_POINT</td>
<td>Indicates that the extension is being called from the script /Radius/Services/&lt;service&gt;/OutgoingScript, when passed to extensions used as scripts.</td>
</tr>
</tbody>
</table>
Java Attribute Dictionary

Appendix A Cisco Prime Access Registrar Tcl, REX, and Java Dictionaries

Table A-11 Marker Extension Interface Variables (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
</table>
| EXT_OUTGOING_CLIENT_SCRIPTING_POINT | static final int
EXT_OUTGOING_CLIENT_SCRIPTING_POINT | Indicates that the extension is being called from the script
/RADIUS/Clients/<client>/OutgoingScript or from the script
/RADIUS/RemoteServers/<server>/OutgoingScript, when passed to extensions used as scripts. |
| EXT_OUTGOING_VENDOR_SCRIPTING_POINT | static final int
EXT_OUTGOING_VENDOR_SCRIPTING_POINT | Indicates that the extension is being called from the script
/RADIUS/Vendors/<vendor>/OutgoingScript, when passed to extensions used as scripts. |
| EXT_OUTGOING_SERVER_SCRIPTING_POINT | static final int
EXT_OUTGOING_SERVER_SCRIPTING_POINT | Indicates that the extension is being called from the script
/RADIUS/OutgoingScript, when passed to extensions used as scripts. |
| EXT_REMOTE_SERVER_OUTAGE_SCRIPTING_POINT | static final int
EXT_REMOTE_SERVER_OUTAGE_SCRIPTING_POINT | Indicates that the extension is being called from the script
/RADIUS/Services/<service>/OutageScript, when passed to extensions used as scripts. |
| EXT_INCOMING_SESSIONMANAGER_SCRIPTING_POINT | static final int
EXT_INCOMING_SESSIONMANAGER_SCRIPTING_POINT | Indicates that the extension is being called from the script
/RADIUS/SessionManagers/<sessionmgr>/IncomingScript, when passed to extensions used as scripts. |
| EXT_OUTGOING_SESSIONMANAGER_SCRIPTING_POINT | static final int
EXT_OUTGOING_SESSIONMANAGER_SCRIPTING_POINT | Indicates that the extension is being called from the script
/RADIUS/SessionManagers/<sessionmgr>/OutgoingScript, when passed to extensions used as scripts. |

Class Sessionrecord

Each request processed by an Extension will have a corresponding session. The methods present in this class operate on the attributes cached in that session record. Group of attributes are cached as an AttributeDictionary in the session record.

Session Record Methods

Table A-12 lists the methods you can use for Session record.

Table A-12 Session Record Methods

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>public java.lang.String get(java.lang.String key)</td>
<td>Returns the first value associated with key.</td>
</tr>
<tr>
<td>get</td>
<td>public java.lang.String get(java.lang.String key,int index)</td>
<td>Returns the value at position index associated with key.</td>
</tr>
</tbody>
</table>
### Table A-12  Session Record Methods (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>put</td>
<td>public boolean put(java.lang.String key, java.lang.String value)</td>
<td>Associates key with value and stores it to the session record. Any existing values associated with key are removed before adding this association. The value can be retrieved by calling the get method with a key that is equal to the original key.</td>
</tr>
<tr>
<td>put</td>
<td>public boolean put(java.lang.String key, java.lang.String value, int index)</td>
<td>Associates key with value depending on the value of index and stores it in the session record. If index equals ExtensionForSession.EXT_REPLACE, any existing values are removed before adding this new association. If index equals ExtensionForSession.EXT_APPEND, the new value is added at the end of the list of existing values. If index equals ExtensionForSession.EXT_AUGMENT, the new association is only made if the session record does not already have an entry for key. If index is a number greater than or equal to 0 and less than the number of entries in the list, the value is inserted at that position in the list. Otherwise, the value is appended at the end of the list. The value can be retrieved by calling the get method with a key that is equal to the original key and the appropriate index.</td>
</tr>
<tr>
<td>remove</td>
<td>public boolean remove(java.lang.String key)</td>
<td>Removes key (and all corresponding values) from the session record. This method does nothing if key is not in the session record.</td>
</tr>
<tr>
<td>remove</td>
<td>public boolean remove(java.lang.String key, int index)</td>
<td>Removes value at the position index that is associated with key. If the index equals ExtensionForSession.EXT_REMOVE_ALL or if the value being removed is the last value associated with key, the key is removed from the session record. This method does nothing if key is not in the session record.</td>
</tr>
</tbody>
</table>

**Note**

A sample java script is available in the following path “/cisco-ar/examples/java” after the installation of AR.
Environment Dictionary

This appendix describes the environment variables the scripts use to communicate with Cisco Prime Access Registrar (Prime Access Registrar) or to communicate with other scripts.

Prime Access Registrar sets the arguments variable in the Environment dictionary, before calling the InitEntryPoint of each script. The arguments variable is set to the value of the InitEntryPointArgs property corresponding to that script, and it allows the administrator to pass (possibly unique) information to each script initialization function.

Environment variables that are set and read for resource management override provide scripts further control over session management. These environment variables, including the following Acquire-User-Session-Limit, Acquire-Group-Session-Limit, Acquire-IP-Dynamic, Acquire-IP-Per-NAS-Port, Acquire-IPX-Dynamic, and Acquire-USR-VPN, can be set at any point before session management is invoked. These environment variables are read as the packet flows through each Resource Manager that the chosen Session Manager calls. The default setting for these environment variables is TRUE. See the “Resource Managers” section on page 4-38 for additional information about Resource Managers.

This appendix has the following major sections:

- **Cisco Prime Access Registrar Environment Dictionary Variables**
  This section lists environment variables you can use in scripts to communicate with Prime Access Registrar or to communicate with other scripts.

- **Internal Variables**
  This section lists environment variables used by the Prime Access Registrar server for internal operations. The environment variables listed in this section must not be modified by scripts.

### Cisco Prime Access Registrar Environment Dictionary Variables

The following variables are text strings stored in the Environment dictionary passed to each scripting point.

**Accepted-Profiles**

Accepted-Profiles is read during authorization after calling server and client incoming scripts (not set by Prime Access Registrar code). If set, the authorization done by local user lists checks to see if the given user's profile as specified in the user record is one of those in the separated list of profiles. If it is not in the separated list of profiles, the request is rejected.
Accounting-Service

Accounting-Service is set after calling server and client incoming scripts and is used to determine which accounting service is used for this request. If set, the server directs the request to be processed by the specified accounting service.

When Accounting-Service is not set, the DefaultAccountingService (as defined in the server configuration) is used instead.

Acquire-Dynamic-DNS

Acquire-Dynamic-DNS is set and read for resource management override. Acquire-Dynamic-DNS is set to FALSE to skip DNS updating during resource management processing.

Acquire-Group-Session-Limit

Acquire-Group-Session-Limit is set and read for resource management override.

Acquire-Group-Session-Limit is set to FALSE to override the use of group session limit resource management.

Acquire-Home-Agent

Acquire-Home-Agent is set and read for resource management override.

Acquire-Home-Agent is set to FALSE to override the allocation of the home agent IP address during resource management processing.

Acquire-IP-Dynamic

Acquire-IP-Dynamic is set and read for resource management override.

Acquire-IP-Dynamic is set to FALSE to override the use of a managed pool of IP addresses resource management.

Acquire-IPX-Dynamic

Acquire-IPX-Dynamic is set and read for resource management override.

Acquire-IPX-Dynamic is set to FALSE to override the use of a managed pool of IPX addresses resource management.

Acquire-IP-Per-NAS-Port

Acquire-IP-Per-NAS-Port is set and read for resource management override.

Acquire-IP-Per-NAS-Port is set to FALSE to override the use of ports associated with specific IP addresses resource management.
Acquire-Subnet-Dynamic

*Acquire-Subnet-Dynamic* is not always used. If set to FALSE, subnet-dynamic resource managers are skipped.

Acquire-User-Session-Limit

*Acquire-User-Session-Limit* set and read for resource management override. *Acquire-User-Session-Limit* is set to FALSE to override the use of user session limit resource management.

Acquire-USR-VPN

*Acquire-USR-VPN* is set and read for resource management override. *Acquire-USR-VPN* is set to FALSE to override the use of Virtual Private Networks (VPNs) that use USR NAS Clients resource management.

Allow-Null-Password

*Allow-Null-Password* is read during password matching and set in local userlist password matching if not set prior. If *Allow-Null-Password* is set to TRUE, the Prime Access Registrar server accepts requests with null passwords.

Authentication-Service

*Authentication-Service* is set and read for authentication service selection and is used to determine which service is used to authenticate the user. If set, the server directs the request to be processed by the specified authentication service. When *Authentication-Service* is not set, the *DefaultAuthenticationService* is used instead.

Authorization-Service

*Authorization-Service* is set and read for authorization service selection and is used to determine which service to use to authorize the user. If set, the server directs the request to be processed by the specified authorization service. When *Authorization-Service* is not set, the *DefaultAuthorizationService* is used instead.

AuthorizationInfo

The MSISDN information is copied to *AuthorizationInfo* that is fetched by M3UA service.
Backin{\textit{g}g}Store-Env-Vars

\textbf{Backin{\textit{g}}Store-Env-Vars} overrides the BackingStoreEnvironmentVariables property of remote servers of type \textit{odbc-accounting} only when the property BufferAccountingPackets is set to TRUE. The value is a comma separated list of environment variables to be stored along with the packet contents in the local disk.

Blacklisted-IMSI

This variable is configured on a SIGTRAN-M3UA remote server. For any incoming request with an IMSI value, if the variable is set as TRUE, then that IMSI value is blacklisted and will not forwarded to the HLR. For more information, see Blacklisting IMSI Values, page 23-13.

Broadcast-Accounting-Packet

If set to TRUE, \textbf{Broadcast-Accounting-Packet} enables broadcasting of Accounting-on or Accounting-off packets to all remote servers of type \textit{radius}.

Cache-Attributes-In-Session

\textbf{Cache-Attributes-In-Session} is set and read for resource management override. \textbf{Cache-Attributes-In-Session} is set to FALSE to override the caching of attributes by the \textit{session-cache} type of resource manager.

Current-Group-Count

\textbf{Current-Group-Count} is set and read for group session management. If set, the group-session-limit resource manager sets \textbf{Current-Group-Count} to be the new value of the group-session-limit counter.

Cache-Outer-Identity

\textbf{Cache-Outer-Identity value} is set to enable identifying session of an user. If it is set to TRUE, WiMAX session manager will cache the outer identity. If it is set to FALSE, the WiMAX session manager will cache the inner identity. The value is set to FALSE by default.

Destination-IP-Address

\textbf{Destination-IP-Address} is a read only value which is set to the receiver IP address. \textbf{Destination-IP-Address} contains the IP address of the request packet receiver.

Destination-Port

\textbf{Destination-port} is a read only value which is set to the receiving port number. \textbf{Destination-port} contains the port number of the receiver of the request.
Dest-Translation-Type

**Dest-Translation-Type** is configured through the GlobalTitleTranslationScript. When the RoutingIndicator is set to **RTE_GT**, Prime Access Registrar server reads the value that is set in Dest-Translation-Type and sets the TranslationType field of the Called Party Address. The value in this environment variable overrides the value that is configured in the DestinationGTAddress/DestTranslationType property of a remote server, SIGTRAN-M3UA.

Dest-Numbering-Plan

**Dest-Numbering-Plan** is configured through the GlobalTitleTranslationScript. When the RoutingIndicator is set to **RTE_GT**, Prime Access Registrar server reads the value that is set in Dest-Numbering-Plan and sets the NumberingPlan field of the Called Party Address. The value in this environment variable overrides the value that is configured in the DestinationGTAddress/Dest-Numbering-Plan property of a remote server, SIGTRAN-M3UA.

The following are the only values that are used for Dest-Numbering-Plan environment variable:

- DATA
- GENERIC
- ISDN
- ISDNMOB
- LANMOB
- MARMOB
- NWSPEC
- TEL
- TELEX
- UNKN

If you set any variable other than the above ones, Prime Access Registrar server sets the NumberingPlan that is configured in DestinationGTAddress/Dest-Numbering-Plan property of a remote server of type SIGTRAN-M3UA.

Dest-Encoding-Scheme

**Dest-Encoding-Scheme** is configured through the GlobalTitleTranslationScript. When the RoutingIndicator is set to **RTE_GT**, Prime Access Registrar server reads the value that is set in Dest-Encoding-Scheme environment variable and sets the EncodingScheme field of the Called Party Address. The value in this environment variable overrides the value that is configured in the DestinationGTAddress/DestEncodingScheme property of a remote server, SIGTRAN-M3UA.

The following are the only values that are used for Dest-Encoding-Scheme environment variable:

- BCDEVEN
- BCDODD

If you set any variable other than the above ones, Prime Access Registrar server sets the EncodingScheme that is configured in the DestinationGTAddress/DestEncodingScheme property of a remote server of type SIGTRAN-M3UA.
Dest-Nature-Of-Address

Dest-Nature-Of-Address is configured through the GlobalTitleTranslationScript. When the RoutingIndicator is set to RTE_GT, Prime Access Registrar server reads the value that is set in Dest-Nature-Of-Address environment variable and sets the NatureOfAddress field of the Called Party Address. The value in this environment variable overrides the value that is configured in the DestinationGTAddress/ DestNatureofAddress property of a remote server, SIGTRAN-M3UA.

The following are the only values that are used for Dest-Nature-Of-Address environment variable:

- ADDR_NOTPRSNT
- INTNUM
- NATSIGNUM
- SUBNUM

If you set any variable other than the above ones, Prime Access Registrar server sets the NatureOfAddress that is configured in the DestinationGTAddress/ DestNatureofAddress property of a remote server of type SIGTRAN-M3UA.

Dest-GT-Format

Dest-GT-Format configured through the GlobalTitleTranslationScript. When the RoutingIndicator is set to RTE_GT, Prime Access Registrar server reads the value that is set in Dest-GT-Format environment variable and uses this format specified for the Global Title Digits(Address Information).

The value in this environment variable overrides the value that is configured in the DestinationGTAddress/ DestGTFormat property of a remote server, SIGTRAN-M3UA.

The following are the only values that are used for Dest-GT-Format environment variable:

- GTFRMT_0
- GTFRMT_1
- GTFRMT_2
- GTFRMT_3
- GTFRMT_4
- GTFRMT_5

If you set any variable other than the above ones, Prime Access Registrar server sets the GTFormat that is configured in the DestinationGTAddress/ DestGTFormat property of a remote server of type SIGTRAN-M3UA.

Diameter-Application-Id

Diameter-Application-Id is set to get the application ID in the Diameter packet.

Diameter-Command-Code

Diameter-Command-Code is set to get the command codes in the Diameter packet.
**Disable-Accounting-On-Off-Broadcast**

If set to TRUE, **disable-accounting-on-off-broadcast** disables broadcasting of Accounting-On and Accounting-Off packets to all remote servers of type 'radius'.

**DSA-Response-Cache**

DSA-Response-Cache is used while performing DSA (Dynamic Service Authorization) feature in Prime Access Registrar. It is FALSE by default, which will clear the response dictionary before Re-Authentication. If DSA-Response-Cache is set to TRUE, Prime Access Registrar will not clear the response dictionary before Re-Authenticating with next service configured.

**Dynamic-DNS-HostName**

**Dynamic-DNS-HostName** is read while constructing the forward hostname during resource management processing to update DNS entries. If set, the name will be used as forward hostname instead of constructing one.

**Dynamic-Search-Filter**

**Dynamic-Search-Filter** overrides the Filter property in remote servers of type ldap. The format of the value set for **Dynamic-Search-Filter** should be similar to that of the Filter property.

**Dynamic-Search-Path**

**Dynamic-Search-Path** is read for LDAP searching. If set, the server uses it as its LDAP search path rather than the value set in the remote server configuration.

**Dynamic-Search-Scope**

**Dynamic-Search-Scope** is used to dynamically set the SearchScope property of an LDAP remote server configuration on a per-packet basis.

**Dynamic-Service-Loop-Limit**

**Dynamic-Service-Loop-Limit** variable is used to change loop counts. When using the same service for reauthentication and reauthorization, a loop can occur in these services. The loop count, by default is 10. You can change the loop count using this variable.

**Dynamic-User-Password-Attribute**

**Dynamic-User-Password-Attribute** is read for LDAP authentication and overrides the UserPasswordAttribute. If set, the server uses it to retrieve the password field as its LDAP UserPassword attribute instead of the value set in the remote server configuration.
EAP-Actual-Identity

**EAP-Actual-Identity** is a read-only variable that contains the International Mobile Subscriber Identity (IMSI) of the user after a successful EAP-SIM authentication.

EAP-Authentication-Mode

**EAP-Authentication-Mode** is a read-only variable, set after a successful EAP-SIM authentication, that indicates whether the EAP-SIM authentication was a reauthentication or a full authentication.

Enforce-Traffic-Throttling

By default, the value is set to FALSE. When set to TRUE, the traffic throttling check for the packet will be executed.

FetchAuthorizationInfo

When set to TRUE, this variable fetches MSISDN value from the HLR. Do not use **FetchAuthorizationInfo** for authorization. We recommend that you use the authorization service of m3ua instead.

Generate-BEK

Generate-BEK is read when WiMax provisioning service is enabled. If this is set, Prime Access Registrar will generate the Bootstrap Encryption Key in the WiMax flow.

Group-Session-Limit

**Group-Session-Limit** is set and read for group session management. The group-session-limit resource manager sets this environment variable to be the limit of the group-session-limit counter as set by the configuration.

HLR-GlobalTitle-Address

**HLR-GlobalTitle-Address** is configured through the GlobalTitleTranslationScript. When the RoutingIndicator is set to **RTE_GT** in SIGTRAN-M3UA remote server, Prime Access Registrar server reads the value that is set in HLR-GlobalTitle-Address and sets the Destination GT Digits(Address Information field) of the Called Party Address.

HLR-GlobalTitle-Cached

**HLR-GlobalTitle-Cached** is set as TRUE to indicate the HLR GT is cached.
The Home Location Registry (HLR) Global Title address (GT address in calling party address (CgPA)) from the SendAuthenticationInfo (SAI) response is cached and used for subsequent authorization request. This cached HLR GT is added to the environment dictionary of the packet to be available for the authorization flow.

The cached HLR GT overrides both the configured destination GT values and GT script provided GT values. The HLR GT caching works by default for RTE_GT. The cached HLR GT can be overridden by updating the environment variable HLR-GlobalTitle-Cached to FALSE (or anything other than TRUE) in the GT script.

This HLR GT will not be cached for:

- reauthentication flow
- authorize only flow when authentication vectors are already available in cache (as there will not be SAI request.

**HLR-Translated-IMSI**

*HLR-Translated-IMSI* is configured through the IMSITranslationScript. Prime Access Registrar server reads the value in HLR-Translated-IMSI and sets the value as IMSI before sending the request to STP/HLR. The value that is configured in the HLR-Translated-IMSI environment variable overrides the IMSI received in EAP-AKA/EAP-SIM request packet.

**Ignore-Accounting-Signature**

*Ignore-Accounting-Signature* is set after calling server and client incoming scripts and is used to ignore missing or incorrect accounting signatures from NASs. If set, Prime Access Registrar does not check whether the account request packet has been signed with the same shared secret as the NAS.

*Ignore-Accounting-Signature* is used to work with RADIUS implementations that did not sign Accounting-Requests. A script was provided in the distribution (for USR NASs) that could be set in the IncomingScript extension point for the USR Vendor that simply set this environment variable.

**IMSI**

International Mobile System Identifier (IMSI) that is fetched from the response from HLR.

**Incoming-Translation-Groups**

*Incoming-Translation-Groups* is read for authentication while processing responses from a remote RADIUS server. If set, *Incoming-Translation-Groups* specifies the translation groups to be used to filter attributes on requests.

**Master-URL-Fragment**

Used with the Windows Provisioning Service feature, *Master-URL-Fragment* specifies the fragment within the Master URL to be sent back to the provisioning server. *Master-URL-Fragment* can be set to any of the following four values: *signup*, *renewal*, *passwordchange*, and *forceupdate*. If *Master-URL-Fragment* is not set and is required to send the URL, *signup* will be sent by default.
The environmental variable `Send-PEAP-URL-TLV` indicates whether or not to send the URL.

**Misc-Log-Message-Info**

`Misc-Log-Message-Info` is read for packet event logging. If a log message is generated, the value of `Misc-Log-Message-Info` is inserted into the middle of the log message.

**MSISDN**

The Mobile Subscriber ISDN Number (MSISDN) that is fetched from the response from HLR.

**Notification-Code**

The Notification-Code variable is set up to indicate the reason for an authentication or authorization failure for EAP-SIM, EAP-AKA, and EAP-AKA’ services. The common authorization/authentication failure reasons as received from the HLR are:

- Unknown subscriber
- System failure
- Data missing
- Unexpected data value
- Reject / Return with unknown error

Prime Access Registrar reads this environment variable and sends an appropriate message to the client.

**Notification-Service**

Notification-Service is an authorization service and is used to send a notification code to the client in case of authorization failure.

This can be any of the services configured under `/radius/services/` except eap services, accounting services, radius-session, radius-query, and diameter.

**Outgoing-Translation-Groups**

`Outgoing-Translation-Groups` is read while proxying to a remote radius server. If set, `Outgoing-Translation-Groups` specifies the translation groups to be used to filter attributes.

**Pager**

The `aregcmd` command supports the `Pager` environment variable. When the `aregcmd` command `stats` is used and the `Pager` environment variable is set, the output of the `stats` command is displayed using the program specified by the `Pager` environment variable.
PoD/CoA

The PoD/CoA variable is set and read for the CoA/PoD packet processing and used to determine whether the incoming PoD/CoA request must be translated to the corresponding PoD/CoA request. Table B-1 lists the PoD/CoA values and the corresponding functions.

Table B-1 PoD/CoA Values and Functions

<table>
<thead>
<tr>
<th>PoD/CoA Value</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>PoD-CoA</td>
<td>Converts Disconnect-Request to CoA</td>
</tr>
<tr>
<td>CoA-PoD</td>
<td>Converts CoA-Request to PoD</td>
</tr>
<tr>
<td>PoDACK-CoAACK</td>
<td>Converts the Disconnect-ACK to CoA-ACK</td>
</tr>
<tr>
<td>CoAACK-PoDACK</td>
<td>Converts the CoA-ACK to Disconnect-ACK</td>
</tr>
<tr>
<td>PoDNAK-CoANAK</td>
<td>Converts the Disconnect-NAK to CoA-NAK</td>
</tr>
<tr>
<td>CoANAK-PoDNAK</td>
<td>Converts the CoA-NAK to Disconnect-NAK</td>
</tr>
</tbody>
</table>

Query-Service

The Query-Service variable is set and read for the radius-query service selection type. The Query-Service variable must be set before authentication phase begins at the server, vendor, or client incoming scripting point or using the policy engine. If set, the server directs requests to be processed by the specified radius-query service. After the Query-Service variable is set, no AAA processing will be done.

Re-Accounting-Service

Re-Accounting-Service is configured, through script, for dynamic service authorization. When the Re-Accounting-Service is set, the server directs the request to the specified reaccounting service for processing.

Re-Authentication-Service

Re-Authentication-Service is configured, through script, for dynamic service authorization. When the Re-Authentication-Service is set, the server directs the request to the specified reauthentication service for processing.

Re-Authorization-Service

Re-Authorization-Service is configured, through script, for dynamic service authorization. When the Re-Authorization-Service is set, the server directs the request to the specified reauthorization service for processing.
Reject-Reason

Reject-Reason is set when a request is being rejected and contains the Reject-Reason. Prime Access Registrar uses the value of Reject-Reason to look up the reject reason in the reply message table.

If Reject-Reason is set to one of: UnknownUser, UserNotEnabled, UserPasswordInvalid, UnableToAcquireResource, ServiceUnavailable, InternalError, MalformedRequest, ConfigurationError, IncomingScriptFailed, OutgoingScriptFailed, IncomingScriptRejectedRequest, OutgoingScriptRejectedRequest, or TerminationAction, then the value set in the configuration under /Radius/Advanced/ReplyMessages will be returned.

Remote-Server

Remote-Server is set and read for logging a rejected packet from a remote server. Remote-Server records the name and IP address of the remote server to which the request has been forwarded.

Remove-Session-On-Acct-Stop

When set to TRUE, server removes the session on receiving an accounting stop packet.

Remote-Servers-Tried

Remote-Servers-Tried contains a list of remote servers that were tried before a request was accepted or rejected (in the case of a Failover multiple remoteserver policy). The list of servers is a comma-separated list of remote server names.

Request-Authenticator

Request-Authenticator is set for every packet upon reception. Getting the Request-Authenticator from a script returns the value of the request authenticator.

Request-Type

Request-Type is set when a request is first received to the type of request, such as one of Access-Request, Access-Accept, Access-Reject, Accounting-Request, Accounting-Response, or Access-Challenge before calling any extension points.

The request contains a string representation of the RADIUS packet type (code). When Prime Access Registrar does not recognize the packet type, it is represented as “Unknown-Packet-Type-<N>”, where <N> is the numeric value of the packet type (for example “Unknown-Packet-Type-9”). The known packet types are listed in Table B-2.
# Appendix B  Environment Dictionary

## Cisco Prime Access Registrar Environment Dictionary Variables

### Table B-2  Request-Type Packets

<table>
<thead>
<tr>
<th>String</th>
<th>Packet Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access-Request</td>
<td>(1)</td>
</tr>
<tr>
<td>Access-Accept</td>
<td>(2)</td>
</tr>
<tr>
<td>Access-Reject</td>
<td>(3)</td>
</tr>
<tr>
<td>Accounting-Request</td>
<td>(4)</td>
</tr>
<tr>
<td>Accounting-Response</td>
<td>(5)</td>
</tr>
<tr>
<td>Access-Challenge</td>
<td>(11)</td>
</tr>
<tr>
<td>Status-Server</td>
<td>(12)</td>
</tr>
<tr>
<td>Status-Client</td>
<td>(13)</td>
</tr>
<tr>
<td>USR-Resource-Free-Request</td>
<td>(21)</td>
</tr>
<tr>
<td>USR-Resource-Free-Response</td>
<td>(22)</td>
</tr>
<tr>
<td>USR-Resource-Query-Request</td>
<td>(12)</td>
</tr>
<tr>
<td>USR-Resource-Query-Response</td>
<td>(24)</td>
</tr>
<tr>
<td>USR-NAS-Reboot-Request</td>
<td>(26)</td>
</tr>
<tr>
<td>USR-NAS-Reboot-Response</td>
<td>(27)</td>
</tr>
<tr>
<td>Ascend-IPA-Allocate</td>
<td>(50)</td>
</tr>
<tr>
<td>Ascend-IPA-Release</td>
<td>(51)</td>
</tr>
<tr>
<td>USR-Enhanced-RADIUS</td>
<td>(254)</td>
</tr>
</tbody>
</table>

**Note**: **Request-Type** is to be used as a read-only variable by scripts.

## Require-User-To-Be-In-Authorization-List

*Require-User-To-Be-In-Authorization-List* is read for authorization. If we are authorizing with a different service than we authenticated with (not usually done) and the user is not known by the authorization service, the default is to continue on unless this environment variable is set, in which case we reject the request with a cause of Unknown-user.

## Response-Type

*Response-Type* is set and read throughout processing and used to determine whether the request should be accepted, rejected, or challenged. When *Response-Type* is set to “Access-Reject at any time during the processing of a request, no more processing of the request is done, and an Access-Reject response is sent. For other valid values for *Response-Type*, see Table B-2.
Retrace-Packet

If set, **Retrace-Packet** causes a trace of the packet to be displayed during the incoming and outgoing scripts. If set, will cause a second trace of the request packet's contents after running all the incoming scripts and/or a second trace of the response packet's contents before running the outgoing scripts.

Send-PEAP-URI-TLV

When set to TRUE, the URI PEAP-TLV is included along with the Result PEAP-TLV in the access-challenge packet. The authenticating user service (of type userlist, LDAP, or WDA) can set this to TRUE using an extension point script or attribute mapping so that the PEAP-v0 service can send the URI PEAP-TLV. The default value for this is FALSE.

**Note**

This variable is used with the Windows Provisioning Service (WPS) feature.

Session-Key

**Session-Key** is read for session management. If set, the server uses it as the key to look up the session associated with the current request, if any. If not set, the server uses the NAS IP Address and NAS Port to create a session key.

Session-Manager

**Session-Manager** is read after user authorization and determines which dynamic resources to allocate for this user, when one is needed. If set, the server directs the request to be processed by the specified session manager. When not set, the SessionManager (as defined in **DefaultSessionManager**) is used when needed.

Session-Notes

**Session-Notes** is a comma-separated list set to make session information available to scripts. **Session-Notes** contains the names of other environment variables. If set, these variables are stored on a Session as notes.

Session-Service

**Session-Service** is set and read during session management. If set, the server will direct the request to be processed by the specified session service.
**Set-Session-Mgr-And-Key-Upon-Lookup**

When `Set-Session-Mgr-And-Key-Upon-Lookup` is set to TRUE, a session-cache resource manager sets the session-manager and session-key environment variable during a query-lookup, and the Prime Access Registrar server does not cache the response dictionary attributes. `Set-Session-Mgr-And-Key-Upon-Lookup` is set to TRUE by a query-service IncomingScript.

**Skip-Session-Management**

When set to TRUE in a request, `Skip-Session-Management` causes session management to be skipped for the request, even if session management might normally occur.

**Skip-Overriding-Username-With-LDAP-UID**

Skip-Overriding-Username-With-LDAP-UID is used to decide if the username should be replaced with the UID from the LDAP server. When Skip-Overriding-Username-With-LDAP-UID is set to TRUE, the username is not replaced with the UID from the LDAP server.

You can use Skip-Overriding-Username-With-LDAP-UID to retain case sensitivity in usernames when the username given logging into the network is in a different case that the UID in the LDAP server database, such as `User1` and `user1`.

**Skip-Overriding-UserName-With-PEAPIdentity**

Skip-Overriding-UserName-With-PEAPIdentity is used to decide if the username should be replaced with the PEAP Identity. When Skip-Overriding-UserName-With-PEAPIdentity is set to TRUE, the username is not replaced with the PEAP Identity.

**Source-IP-Address**

`Source-IP-Address` is set when a request is first received to the IP address from which the IP request was received before calling any extension points. `Source-IP-Address` contains the IP address of the NAS or proxy server that sent the request to this server.

**Note**

`Source-IP-Address` is to be used as a read-only variable by scripts.

**Source-Port**

`Source-Port` is set when a request is first received to the port from which the request was received. `Source-Port` is set for each request before calling any extension points and contains the port on the NAS or proxy server that was used to send the request to this server.

**Note**

`Source-Port` is to be used as a read-only variable by scripts.
Appendix B      Environment Dictionary

Cisco Prime Access Registrar Environment Dictionary Variables

**SQL-Sequence**

The **SQL-Sequence** variable is set with a list of SQL statement names, separated by a semicolon (;). For example, the SQL statement names ‘sql3’, ‘sql4’, and ‘sql5’ are denoted as sql3;sql4;sql5. If the variable is set, Prime Access Registrar picks the SQL statements and executes them in the order specified.

**Subnet-Size-If-No-Match**

The **Subnet-Size-If-No-Match** is set to one of BIGGER, SMALLER or EXACT, determines the behavior of the subnet-dynamic resource manager if a pool of the requested size is not available.

**Trace-Level**

The **Trace-Level** is set for each request before calling any extension points. **Trace-Level** is set to the current trace level as specified through `aregcmd`. If set by a script, **Trace-Level** changes the trace level used to determine what level of information is traced.

**Unavailable-Resource**

The **Unavailable-Resource** is set during session management. If the request is being rejected because one of the resource managers failed to allocate a resource, **Unavailable-Resource** is set to the name of the resource manager that failed.

**Unavailable-Resource-Type**

The **Unavailable-Resource-Type** is set during session management. If the request is being rejected because one of the resource managers failed to allocate a resource, **Unavailable-Resource-Type** is set to the type of the resource manager that failed.

**UserDefined1**

The **UserDefined1** is set to the value of the UserDefined1 property of the user from a local user list during password matching of local users.

**User-Authorization-Script**

The **User-Authorization-Script** is read in local services during authorization. If set, the server calls the specified script to do additional user authorization after authentication succeeds.

**User-Group**

The **User-Group** is read in local services during authorization. If set, specifies the UserGroup to which the current user belongs.
User-Group-Session-Limit

User-Group-Session-Limit is read during session management. If set, User-Group-Session-Limit overrides the limit specified for the group-session-limit resource manager.

User-Name

User-Name is read by a local service during authentication. When User-Name is set, it is the name used to authenticate or authorize the request and overrides the User-Name in the Request dictionary.

User-Profile

User-Profile is read in local services during authorization. If set, User-Profile specifies the Profile from which the current user should receive attributes.

User-Session-Limit

User-Session-Limit is read during session management. If set, User-Session-Limit overrides the limit specified for the user-session-limit resource manager.

Virtual-Server-Outgoing-Script

Virtual-Server-Outgoing-Script is read when LawfulIntercept script object is enabled to use virtual script object. If this is set, the configured script will be executed after server outgoing script.

X509- Subject-Name

X509- Subject-Name reads the value of the subject in the SSL certificate. This is read while processing the access request.

Internal Variables

The following environment variables are used by the server for internal operation. The values for these environment variables must not be modified.

- Add-Message-Authenticator
- Calling-Service-Name
- Cleartext-Password
- Current-Service-Name
- Dynamic-Search-UID
- Duplicate-Req
- EAP-Internal-Services
• Group-Service
• Group-Service-State-ID
• Hidden-Attrib
• IMSI
• Local-Port-type
• Message-Authenticator-Present
• MSCHAP-Account-Name
• MS-ChapV2-Message
• NAS-Name-And-IPAddress
• Notify-Service-Session-Key
• Notify-Service-State-ID
• Number-Requested-Quintets
• Number-Requested-Triplets
• Proxied-Dynamic-Auth (named Proxied-POD in earlier releases)
• Provider-Identifier
• Rcd-NT-Password-Hash-Hash (named Rcd-NT-Password-Hash in earlier releases)
• Remote-Session
• Return-Data
• Roaming
• Script-Level
• Session-ID
• Session-Accounting-Counter
• Session-Generation-Tag
• Session-Last-Accessed-Time
• Session-Manager-Key
• Session-NAS-Identifier
• Session-NAS-Port
• Session-Resource-Count
• Session-Resource-%d
• Session-Reuse
• Session-Start-Time
• Session-Survives-NAS-Reboot
• Session-User-Name
• User-Name-Used-For-Lookup
• WiMax-Authentication
• WiMax-SessionManager-Exists
RADIUS Attributes

This appendix lists the attributes Cisco Prime Access Registrar (Prime Access Registrar) supports with their names and values. RADIUS attributes carry the specific authentication, authorization information, and configuration details for requests and replies. For more detailed information about specific attributes, see the appropriate RFC as listed Table C-1.

Table C-1  RFCs for RADIUS Attributes

<table>
<thead>
<tr>
<th>RFC Subject</th>
<th>RFC Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard RADIUS Attributes</td>
<td>2865</td>
</tr>
<tr>
<td>RADIUS Accounting Attributes</td>
<td>2866</td>
</tr>
<tr>
<td>Accounting Modifications for Tunnel Protocol Support</td>
<td>2867</td>
</tr>
<tr>
<td>Attributes for Tunnel Protocol Support</td>
<td>2868</td>
</tr>
<tr>
<td>RADIUS Extensions</td>
<td>2869</td>
</tr>
</tbody>
</table>

This appendix has two sections:

- **RADIUS Attributes**—This section provides an alphabetic list of all RADIUS attributes Prime Access Registrar supports and a list of all RADIUS attributes in numeric order.
- **Vendor-Specific Attributes**—This section provides lists of RADIUS vendor-specific attributes (VSAs).

RADIUS Attributes

This section lists the RADIUS attributes supported in Prime Access Registrar. RADIUS attributes carry specific authentication, authorization, information, and configuration details in the Access-Request and the RADIUS server response.

Cisco Prime Access Registrar Attributes

Table C-2 provides an alphabetical list of all attributes used in Prime Access Registrar and the attribute number.
### Table C-2 RADIUS Attributes Alphabetical List

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Attribute Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acct-Authentic</td>
<td>45</td>
</tr>
<tr>
<td>Acct-Delay-Time</td>
<td>41</td>
</tr>
<tr>
<td>Acct-Input-Gigawords</td>
<td>52</td>
</tr>
<tr>
<td>Acct-Input-Octets</td>
<td>42</td>
</tr>
<tr>
<td>Acct-Input-Packets</td>
<td>47</td>
</tr>
<tr>
<td>Acct-Interim-Interval</td>
<td>85</td>
</tr>
<tr>
<td>Acct-Link-Count</td>
<td>51</td>
</tr>
<tr>
<td>Acct-Multi-Session-Id</td>
<td>50</td>
</tr>
<tr>
<td>Acct-Output-Gigawords</td>
<td>53</td>
</tr>
<tr>
<td>Acct-Output-Octets</td>
<td>43</td>
</tr>
<tr>
<td>Acct-Output-Packets</td>
<td>48</td>
</tr>
<tr>
<td>Acct-Session-Id</td>
<td>44</td>
</tr>
<tr>
<td>Acct-Session-Time</td>
<td>46</td>
</tr>
<tr>
<td>Acct-Status-Type</td>
<td>40</td>
</tr>
<tr>
<td>Acct-Terminate-Cause</td>
<td>49</td>
</tr>
<tr>
<td>Acct-Tunnel-Connection</td>
<td>68</td>
</tr>
<tr>
<td>Acct-Tunnel-Packets-Lost</td>
<td>86</td>
</tr>
<tr>
<td>Acquire-Group-Session-Limit</td>
<td>280</td>
</tr>
<tr>
<td>ARAP-Challenge-Response</td>
<td>84</td>
</tr>
<tr>
<td>ARAP-Features</td>
<td>71</td>
</tr>
<tr>
<td>ARAP-Password</td>
<td>70</td>
</tr>
<tr>
<td>ARAP-Security</td>
<td>73</td>
</tr>
<tr>
<td>ARAP-Security-Data</td>
<td>74</td>
</tr>
<tr>
<td>ARAP-Zone-Access</td>
<td>72</td>
</tr>
<tr>
<td>Callback-Id</td>
<td>20</td>
</tr>
<tr>
<td>Callback-Number</td>
<td>19</td>
</tr>
<tr>
<td>Called-Station-Id</td>
<td>30</td>
</tr>
<tr>
<td>Calling-Station-Id</td>
<td>31</td>
</tr>
<tr>
<td>Change-Password</td>
<td>17</td>
</tr>
<tr>
<td>CHAP-Challenge</td>
<td>60</td>
</tr>
<tr>
<td>CHAP-Password</td>
<td>3</td>
</tr>
<tr>
<td>Class</td>
<td>25</td>
</tr>
<tr>
<td>Configuration-Token</td>
<td>78</td>
</tr>
<tr>
<td>Connect-Info</td>
<td>77</td>
</tr>
<tr>
<td>Digest-Attributes</td>
<td>207</td>
</tr>
<tr>
<td>Digest-Response</td>
<td>206</td>
</tr>
</tbody>
</table>
### Table C-2: RADIUS Attributes Alphabetical List (continued)

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Attribute Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>EAP-Message</td>
<td>79</td>
</tr>
<tr>
<td>Error-Cause</td>
<td>101</td>
</tr>
<tr>
<td>Event-Timestamp</td>
<td>55</td>
</tr>
<tr>
<td>Filter-Id</td>
<td>11</td>
</tr>
<tr>
<td>Framed-AppleTalk-Link</td>
<td>37</td>
</tr>
<tr>
<td>Framed-AppleTalk-Network</td>
<td>38</td>
</tr>
<tr>
<td>Framed-AppleTalk-Zone</td>
<td>39</td>
</tr>
<tr>
<td>Framed-Compression</td>
<td>13</td>
</tr>
<tr>
<td>Framed-Interface-Id</td>
<td>96</td>
</tr>
<tr>
<td>Framed-IP-Address</td>
<td>8</td>
</tr>
<tr>
<td>Framed-IP-Netmask</td>
<td>9</td>
</tr>
<tr>
<td>Framed-IPV6-Pool</td>
<td>100</td>
</tr>
<tr>
<td>Framed-IPV6-Prefix</td>
<td>97</td>
</tr>
<tr>
<td>Framed-IPV6-Route</td>
<td>99</td>
</tr>
<tr>
<td>Framed-IPX-Network</td>
<td>12</td>
</tr>
<tr>
<td>Framed-MTU</td>
<td>12</td>
</tr>
<tr>
<td>Framed-Pool</td>
<td>88</td>
</tr>
<tr>
<td>Framed-Protocol</td>
<td>7</td>
</tr>
<tr>
<td>Framed-Route</td>
<td>22</td>
</tr>
<tr>
<td>Framed-Routing</td>
<td>10</td>
</tr>
<tr>
<td>Idle-Timeout</td>
<td>28</td>
</tr>
<tr>
<td>Login-IP-Host</td>
<td>14</td>
</tr>
<tr>
<td>Login-IPV6-Host</td>
<td>98</td>
</tr>
<tr>
<td>Login-LAT-Group</td>
<td>36</td>
</tr>
<tr>
<td>Login-LAT-Node</td>
<td>35</td>
</tr>
<tr>
<td>Login-LAT-Port</td>
<td>63</td>
</tr>
<tr>
<td>Login-LAT-Service</td>
<td>34</td>
</tr>
<tr>
<td>Login-Service</td>
<td>15</td>
</tr>
<tr>
<td>Login-TCP-Port</td>
<td>16</td>
</tr>
<tr>
<td>Message-Authenticator</td>
<td>80</td>
</tr>
<tr>
<td>NAS-Identifier</td>
<td>32</td>
</tr>
<tr>
<td>NAS-IP-Address</td>
<td>4</td>
</tr>
<tr>
<td>NAS-IPV6-Address</td>
<td>95</td>
</tr>
<tr>
<td>NAS-Port</td>
<td>5</td>
</tr>
<tr>
<td>NAS-Port-ID</td>
<td>87</td>
</tr>
<tr>
<td>NAS-Port-Type</td>
<td>61</td>
</tr>
</tbody>
</table>
Table C-2 RADIUS Attributes Alphabetical List (continued)

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Attribute Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Originating-Line-Info</td>
<td>94</td>
</tr>
<tr>
<td>Password-Expiration</td>
<td>21</td>
</tr>
<tr>
<td>Password-Retry</td>
<td>75</td>
</tr>
<tr>
<td>Port-Limit</td>
<td>62</td>
</tr>
<tr>
<td>Prompt</td>
<td>76</td>
</tr>
<tr>
<td>Proxy-State</td>
<td>33</td>
</tr>
<tr>
<td>Reply-Message</td>
<td>18</td>
</tr>
<tr>
<td>Service-Type</td>
<td>6</td>
</tr>
<tr>
<td>Session-Timeout</td>
<td>27</td>
</tr>
<tr>
<td>State</td>
<td>24</td>
</tr>
<tr>
<td>Termination-Action</td>
<td>29</td>
</tr>
<tr>
<td>Text-Ascend-Data-Filter</td>
<td>225</td>
</tr>
<tr>
<td>Tunnel-Assignment-ID</td>
<td>82</td>
</tr>
<tr>
<td>Tunnel-Client-Auth-ID</td>
<td>90</td>
</tr>
<tr>
<td>Tunnel-Client-Endpoint</td>
<td>66</td>
</tr>
<tr>
<td>Tunnel-Medium-Type</td>
<td>65</td>
</tr>
<tr>
<td>Tunnel-Password</td>
<td>69</td>
</tr>
<tr>
<td>Tunnel-Preference</td>
<td>83</td>
</tr>
<tr>
<td>Tunnel-Private-Group-ID</td>
<td>81</td>
</tr>
<tr>
<td>Tunnel-Server-Auth-ID</td>
<td>91</td>
</tr>
<tr>
<td>Tunnel-Server-Endpoint</td>
<td>67</td>
</tr>
<tr>
<td>Tunnel-Type</td>
<td>64</td>
</tr>
<tr>
<td>User-Name</td>
<td>1</td>
</tr>
<tr>
<td>User-Password</td>
<td>2</td>
</tr>
<tr>
<td>Vendor-Specific Attributes</td>
<td>26</td>
</tr>
</tbody>
</table>

RADIUS Attributes Numeric List

Table C-3 lists all RFC-defined RADIUS attributes in numeric order.

Table C-3 RADIUS Attributes Numeric List

<table>
<thead>
<tr>
<th>Number</th>
<th>Attribute Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>User-Name</td>
</tr>
<tr>
<td>2</td>
<td>User-Password</td>
</tr>
<tr>
<td>3</td>
<td>CHAP-Password</td>
</tr>
<tr>
<td>4</td>
<td>NAS-IP-Address</td>
</tr>
<tr>
<td>Number</td>
<td>Attribute Name</td>
</tr>
<tr>
<td>--------</td>
<td>----------------------</td>
</tr>
<tr>
<td>5</td>
<td>NAS-Port</td>
</tr>
<tr>
<td>6</td>
<td>Service-Type</td>
</tr>
<tr>
<td>7</td>
<td>Framed-Protocol</td>
</tr>
<tr>
<td>8</td>
<td>Framed-IP-Address</td>
</tr>
<tr>
<td>9</td>
<td>Framed-IP-Netmask</td>
</tr>
<tr>
<td>10</td>
<td>Framed-Routing</td>
</tr>
<tr>
<td>11</td>
<td>Filter-Id</td>
</tr>
<tr>
<td>12</td>
<td>Framed-MTU</td>
</tr>
<tr>
<td>13</td>
<td>Framed-Compression</td>
</tr>
<tr>
<td>14</td>
<td>Login-IP-Host</td>
</tr>
<tr>
<td>15</td>
<td>Login-Service</td>
</tr>
<tr>
<td>16</td>
<td>Login-TCP-Port</td>
</tr>
<tr>
<td>17</td>
<td>Change-Password</td>
</tr>
<tr>
<td>18</td>
<td>Reply-Message</td>
</tr>
<tr>
<td>19</td>
<td>Callback-Number</td>
</tr>
<tr>
<td>20</td>
<td>Callback-Id</td>
</tr>
<tr>
<td>21</td>
<td>Password-Expiration</td>
</tr>
<tr>
<td>22</td>
<td>Framed-Route</td>
</tr>
<tr>
<td>23</td>
<td>Framed-IPX-Network</td>
</tr>
<tr>
<td>24</td>
<td>State</td>
</tr>
<tr>
<td>25</td>
<td>Class</td>
</tr>
</tbody>
</table>
### Table C-3  RADIUS Attributes Numeric List (continued)

<table>
<thead>
<tr>
<th>Number</th>
<th>Attribute Name</th>
</tr>
</thead>
</table>
| 26     | Vendor-Specific Attributes (VSAs)  
See Vendor-Specific Attributes, page C-13 or the specific vendor’s VSA list:  
- 3GPP VSAs  
- 3GPP2 VSAs  
- ACC VSAs  
- Altiga VSAs  
- Ascend VSAs  
- Bay Networks VSAs  
- Cabletron VSAs  
- Cisco Prime Access Registrar Internal VSAs  
- Cisco VSAs  
- Compatible VSAs  
- Microsoft VSAs  
- Nomadix VSAs  
- RedBack VSAs  
- RedCreek VSAs  
- Telebit VSAs  
- Unisphere VSAs  
- USR VSAs  
- WiMax  
- WISPr  
- XML |
<p>| 27     | Session-Timeout |
| 28     | Idle-Timeout   |
| 29     | Termination-Action |
| 30     | Called-Station-ID (DNIS) |
| 31     | Calling-Station-ID (CLID) |
| 32     | NAS-Identifier |
| 33     | Proxy-State    |
| 34     | Login-LAT-Service |
| 35     | Login-LAT-Node |
| 36     | Login-LAT-Group |
| 37     | Framed-AppleTalk-Link |</p>
<table>
<thead>
<tr>
<th>Number</th>
<th>Attribute Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>Framed-AppleTalk-Network</td>
</tr>
<tr>
<td>39</td>
<td>Framed-AppleTalk-Zone</td>
</tr>
<tr>
<td>40</td>
<td>Acct-Status-Type</td>
</tr>
<tr>
<td>41</td>
<td>Acct-Delay-Time</td>
</tr>
<tr>
<td>42</td>
<td>Acct-Input-Octets</td>
</tr>
<tr>
<td>43</td>
<td>Acct-Output-Octets</td>
</tr>
<tr>
<td>44</td>
<td>Acct-Session-Id</td>
</tr>
<tr>
<td>45</td>
<td>Acct-Authentic</td>
</tr>
<tr>
<td>46</td>
<td>Acct-Session-Time</td>
</tr>
<tr>
<td>47</td>
<td>Acct-Input-packets</td>
</tr>
<tr>
<td>48</td>
<td>Acct-Output-packets</td>
</tr>
<tr>
<td>49</td>
<td>Acct-Terminate-Cause</td>
</tr>
<tr>
<td>50</td>
<td>Acct-Multi-Session-Id</td>
</tr>
<tr>
<td>51</td>
<td>Acct-Link-Count</td>
</tr>
<tr>
<td>52</td>
<td>Acct-Input-Gigawords</td>
</tr>
<tr>
<td>53</td>
<td>Acct-Output-Gigawords</td>
</tr>
<tr>
<td>54</td>
<td>unassigned</td>
</tr>
<tr>
<td>55</td>
<td>Event-Timestamp</td>
</tr>
<tr>
<td>56</td>
<td>unassigned</td>
</tr>
<tr>
<td>57</td>
<td>unassigned</td>
</tr>
<tr>
<td>58</td>
<td>unassigned</td>
</tr>
<tr>
<td>59</td>
<td>unassigned</td>
</tr>
<tr>
<td>60</td>
<td>CHAP-Challenge</td>
</tr>
<tr>
<td>61</td>
<td>NAS-Port-Type</td>
</tr>
<tr>
<td>62</td>
<td>Port-Limit</td>
</tr>
<tr>
<td>63</td>
<td>Login-LAT-PortNo</td>
</tr>
<tr>
<td>64</td>
<td>Tunnel-Type</td>
</tr>
<tr>
<td>65</td>
<td>Tunnel-Medium-Type</td>
</tr>
<tr>
<td>66</td>
<td>Tunnel-Client-Endpoint</td>
</tr>
<tr>
<td>67</td>
<td>Tunnel-Server-Endpoint</td>
</tr>
<tr>
<td>68</td>
<td>Acct-Tunnel-Connection</td>
</tr>
<tr>
<td>69</td>
<td>Tunnel-ID</td>
</tr>
<tr>
<td>70</td>
<td>Tunnel-Password</td>
</tr>
<tr>
<td>71</td>
<td>ARAP-Password</td>
</tr>
<tr>
<td>72</td>
<td>ARAP-Features</td>
</tr>
<tr>
<td>73</td>
<td>ARAP-Zone-Access</td>
</tr>
<tr>
<td>Number</td>
<td>Attribute Name</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>73</td>
<td>ARAP-Security</td>
</tr>
<tr>
<td>74</td>
<td>ARAP-Security-Data</td>
</tr>
<tr>
<td>75</td>
<td>Password-Retry</td>
</tr>
<tr>
<td>76</td>
<td>Prompt</td>
</tr>
<tr>
<td>77</td>
<td>Connect-Info</td>
</tr>
<tr>
<td>78</td>
<td>Configuration-Token</td>
</tr>
<tr>
<td>79</td>
<td>EAP-Message</td>
</tr>
<tr>
<td>80</td>
<td>Message-Authenticator</td>
</tr>
<tr>
<td>81</td>
<td>Tunnel-Private-Group-ID</td>
</tr>
<tr>
<td>81</td>
<td>Ascend-Auth-Type</td>
</tr>
<tr>
<td>82</td>
<td>Tunnel-Assignment-ID</td>
</tr>
<tr>
<td>83</td>
<td>Tunnel-Preference</td>
</tr>
<tr>
<td>84</td>
<td>ARAP-Challenge-Response</td>
</tr>
<tr>
<td>85</td>
<td>Acct-Interim-Interval</td>
</tr>
<tr>
<td>85</td>
<td>Ascend-IP-Pool-Chaining</td>
</tr>
<tr>
<td>86</td>
<td>Acct-Tunnel-Packets-Lost</td>
</tr>
<tr>
<td>87</td>
<td>NAS-Port-ID</td>
</tr>
<tr>
<td>88</td>
<td>Framed-Pool</td>
</tr>
<tr>
<td>88</td>
<td>Ascend-IP-TOS</td>
</tr>
<tr>
<td>89</td>
<td>Ascend-IP-TOS-Precedence</td>
</tr>
<tr>
<td>90</td>
<td>Tunnel-Client-Auth-ID</td>
</tr>
<tr>
<td>90</td>
<td>Ascend-IP-TOS-Apply-To</td>
</tr>
<tr>
<td>91</td>
<td>Tunnel-Server-Auth-ID</td>
</tr>
<tr>
<td>91</td>
<td>Ascend-Filter</td>
</tr>
<tr>
<td>92</td>
<td>Ascend-Dsl-Rate-Type</td>
</tr>
<tr>
<td>93</td>
<td>Ascend-Redirect-Number</td>
</tr>
<tr>
<td>94</td>
<td>Originating-Line-Info</td>
</tr>
<tr>
<td>95</td>
<td>Ascend-ATM-Vci</td>
</tr>
<tr>
<td>96</td>
<td>Ascend-Source-IP-Check</td>
</tr>
<tr>
<td>97</td>
<td>Ascend-Dsl-Rate-Mode</td>
</tr>
<tr>
<td>98</td>
<td>Ascend-Dsl-Upstream-Limit</td>
</tr>
<tr>
<td>99</td>
<td>Ascend-Dsl-Downstream-Limit</td>
</tr>
<tr>
<td>100</td>
<td>Ascend-Dsl-CIR-Recv-Limit</td>
</tr>
<tr>
<td>101</td>
<td>Error-Cause</td>
</tr>
<tr>
<td>102</td>
<td>EAP-Key-Name</td>
</tr>
<tr>
<td>103</td>
<td>Ascend-Source-Auth</td>
</tr>
<tr>
<td>Number</td>
<td>Attribute Name</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------</td>
</tr>
<tr>
<td>104</td>
<td>Ascend-Private-Route</td>
</tr>
<tr>
<td>105</td>
<td>unassigned</td>
</tr>
<tr>
<td>106</td>
<td>Ascend-FR-Link-Status-DLCI</td>
</tr>
<tr>
<td>107</td>
<td>unassigned</td>
</tr>
<tr>
<td>108</td>
<td>Ascend-Callback-Delay</td>
</tr>
<tr>
<td>109</td>
<td>unassigned</td>
</tr>
<tr>
<td>110</td>
<td>unassigned</td>
</tr>
<tr>
<td>111</td>
<td>Ascend-Multicast-GLeave-Delay</td>
</tr>
<tr>
<td>112</td>
<td>Ascend-CBCP-Enable</td>
</tr>
<tr>
<td>113</td>
<td>Ascend-CBCP-Mode</td>
</tr>
<tr>
<td>114</td>
<td>unassigned</td>
</tr>
<tr>
<td>115</td>
<td>Ascend-CBCP-Trunk-Group</td>
</tr>
<tr>
<td>116</td>
<td>Ascend-Appletalk-Route</td>
</tr>
<tr>
<td>117</td>
<td>Ascend-Appletalk-Peer-Mode</td>
</tr>
<tr>
<td>118</td>
<td>Ascend-Route-Appletalk</td>
</tr>
<tr>
<td>119</td>
<td>unassigned</td>
</tr>
<tr>
<td>120</td>
<td>Ascend-Modem-PortNo</td>
</tr>
<tr>
<td>121</td>
<td>Ascend-Modem-SlotNo</td>
</tr>
<tr>
<td>122</td>
<td>unassigned</td>
</tr>
<tr>
<td>123</td>
<td>unassigned</td>
</tr>
<tr>
<td>124</td>
<td>unassigned</td>
</tr>
<tr>
<td>125</td>
<td>Ascend-Maximum-Call-Duration</td>
</tr>
<tr>
<td>126</td>
<td>Ascend-Preference</td>
</tr>
<tr>
<td>127</td>
<td>Tunneling-Protocol</td>
</tr>
<tr>
<td>128</td>
<td>Ascend-Shared-Profile-Enable</td>
</tr>
<tr>
<td>129</td>
<td>Ascend-Primary-Home-Agent</td>
</tr>
<tr>
<td>130</td>
<td>Ascend-Secondary-Home-Agent</td>
</tr>
<tr>
<td>131</td>
<td>Ascend-Dialout-Allowed</td>
</tr>
<tr>
<td>132</td>
<td>Ascend-Client-Gateway</td>
</tr>
<tr>
<td>133</td>
<td>Ascend-BACP-Enable</td>
</tr>
<tr>
<td>134</td>
<td>Ascend-DHCP-Maximum-Leases</td>
</tr>
<tr>
<td>135</td>
<td>Ascend-Client-Primary-DNS</td>
</tr>
<tr>
<td>136</td>
<td>Ascend-Client-Secondary-DNS</td>
</tr>
<tr>
<td>137</td>
<td>Ascend-Client-Assign-DNS</td>
</tr>
<tr>
<td>138</td>
<td>Ascend-User-Acct-Type</td>
</tr>
<tr>
<td>139</td>
<td>Ascend-User-Acct-Host</td>
</tr>
</tbody>
</table>
### Table C-3  RADIUS Attributes Numeric List (continued)

<table>
<thead>
<tr>
<th>Number</th>
<th>Attribute Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>140</td>
<td>Ascend-User-Acct-Port</td>
</tr>
<tr>
<td>141</td>
<td>Ascend-User-Acct-Key</td>
</tr>
<tr>
<td>142</td>
<td>Ascend-User-Acct-Base</td>
</tr>
<tr>
<td>143</td>
<td>Ascend-User-Acct-Time</td>
</tr>
<tr>
<td>144</td>
<td>Ascend-Assign-IP-Client</td>
</tr>
<tr>
<td>145</td>
<td>Ascend-Assign-IP-Server</td>
</tr>
<tr>
<td>146</td>
<td>Ascend-Assign-IP-Global-Pool</td>
</tr>
<tr>
<td>147</td>
<td>Ascend-DHCP-Reply</td>
</tr>
<tr>
<td>148</td>
<td>Ascend-DHCP-Pool-Number</td>
</tr>
<tr>
<td>149</td>
<td>Ascend-Expect-Callback</td>
</tr>
<tr>
<td>150</td>
<td>Ascend-Event-Type</td>
</tr>
<tr>
<td>151</td>
<td>Ascend-Session-Svr-Key</td>
</tr>
<tr>
<td>152</td>
<td>Ascend-Multicast-Rate-Limit</td>
</tr>
<tr>
<td>153</td>
<td>Ascend-IF-Netmask</td>
</tr>
<tr>
<td>154</td>
<td>Ascend-Remote-Addr</td>
</tr>
<tr>
<td>155</td>
<td>Ascend-Multicast-Client</td>
</tr>
<tr>
<td>156</td>
<td>Ascend-FR-Circuit-Name</td>
</tr>
<tr>
<td>157</td>
<td>Ascend-FR-LinkUp</td>
</tr>
<tr>
<td>158</td>
<td>Ascend-FR-Nailed-Grp</td>
</tr>
<tr>
<td>159</td>
<td>Ascend-FR-Type</td>
</tr>
<tr>
<td>160</td>
<td>Ascend-FR-Link-Mgt</td>
</tr>
<tr>
<td>161</td>
<td>Ascend-FR-N391</td>
</tr>
<tr>
<td>162</td>
<td>Ascend-FR-DCE-N392</td>
</tr>
<tr>
<td>163</td>
<td>Ascend-FR-DTE-N392</td>
</tr>
<tr>
<td>164</td>
<td>Ascend-FR-DCE-N393</td>
</tr>
<tr>
<td>165</td>
<td>Ascend-FR-DTE-N393</td>
</tr>
<tr>
<td>166</td>
<td>Ascend-FR-T391</td>
</tr>
<tr>
<td>167</td>
<td>Ascend-FR-T392</td>
</tr>
<tr>
<td>168</td>
<td>Ascend-Bridge-Address</td>
</tr>
<tr>
<td>169</td>
<td>Ascend-TS-Idle-Limit</td>
</tr>
<tr>
<td>170</td>
<td>Ascend-TS-Idle-Mode</td>
</tr>
<tr>
<td>171</td>
<td>Ascend-DBA-Monitor</td>
</tr>
<tr>
<td>172</td>
<td>Ascend-Base-Channel-Count</td>
</tr>
<tr>
<td>173</td>
<td>Ascend-Minimum-Channels</td>
</tr>
<tr>
<td>174</td>
<td>Ascend-IPX-Route</td>
</tr>
<tr>
<td>175</td>
<td>Ascend-FT1-Caller</td>
</tr>
</tbody>
</table>

Cisco Prime Access Registrar 7.0.1 User Guide
<table>
<thead>
<tr>
<th>Number</th>
<th>Attribute Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>176</td>
<td>Ascend-backup</td>
</tr>
<tr>
<td>177</td>
<td>Ascend-Call-Type</td>
</tr>
<tr>
<td>178</td>
<td>Ascend-Group</td>
</tr>
<tr>
<td>179</td>
<td>Ascend-FR-DLCI</td>
</tr>
<tr>
<td>180</td>
<td>Ascend-FR-Profile-Name</td>
</tr>
<tr>
<td>181</td>
<td>Ascend-Ara-PW</td>
</tr>
<tr>
<td>182</td>
<td>Ascend-IPX-Node-Addr</td>
</tr>
<tr>
<td>183</td>
<td>Ascend-Home-Agent-IP-Addr</td>
</tr>
<tr>
<td>184</td>
<td>Ascend-Home-Agent-Password</td>
</tr>
<tr>
<td>185</td>
<td>Ascend-Home-Network-Name</td>
</tr>
<tr>
<td>186</td>
<td>Ascend-Home-Agent-UDP-Port</td>
</tr>
<tr>
<td>187</td>
<td>Ascend-Multilink-ID supported</td>
</tr>
<tr>
<td>188</td>
<td>Ascend-Num-In-Multilink</td>
</tr>
<tr>
<td>189</td>
<td>Ascend-First-Dest (Not supported)</td>
</tr>
<tr>
<td>190</td>
<td>Ascend-Pre-Input-Octets</td>
</tr>
<tr>
<td>191</td>
<td>Ascend-Pre-Output-Octets</td>
</tr>
<tr>
<td>192</td>
<td>Ascend-Pre-Input-packets</td>
</tr>
<tr>
<td>193</td>
<td>Ascend-Pre-Output-packets</td>
</tr>
<tr>
<td>194</td>
<td>Ascend-Maximum-Time</td>
</tr>
<tr>
<td>195</td>
<td>Ascend-Disconnect-Cause</td>
</tr>
<tr>
<td>196</td>
<td>Ascend-Connect-Progress</td>
</tr>
<tr>
<td>197</td>
<td>Ascend-Data-Rate</td>
</tr>
<tr>
<td>198</td>
<td>Ascend-PreSession-Time</td>
</tr>
<tr>
<td>199</td>
<td>Ascend-Token-Idle</td>
</tr>
<tr>
<td>200</td>
<td>Ascend-Token-Immediate</td>
</tr>
<tr>
<td>201</td>
<td>Ascend-Require-Auth</td>
</tr>
<tr>
<td>202</td>
<td>Ascend-Number-Sessions</td>
</tr>
<tr>
<td>203</td>
<td>Ascend-Authen-Alias</td>
</tr>
<tr>
<td>204</td>
<td>Ascend-Token-Expiry</td>
</tr>
<tr>
<td>205</td>
<td>Ascend-Menu-Selector</td>
</tr>
<tr>
<td>206</td>
<td>Digest-Response</td>
</tr>
<tr>
<td>207</td>
<td>Digest-Attributes</td>
</tr>
<tr>
<td>208</td>
<td>Ascend-PW-Lifetime</td>
</tr>
<tr>
<td>209</td>
<td>Ascend-IP-Direct</td>
</tr>
<tr>
<td>210</td>
<td>Ascend-PPP-VJ-Slot-Comp</td>
</tr>
<tr>
<td>211</td>
<td>Ascend-PPP-VJ-1172</td>
</tr>
<tr>
<td>Number</td>
<td>Attribute Name</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>212</td>
<td>Ascend-PPP-Async-Map</td>
</tr>
<tr>
<td>213</td>
<td>Ascend-Third-Prompt</td>
</tr>
<tr>
<td>214</td>
<td>Ascend-Send-Secret</td>
</tr>
<tr>
<td>215</td>
<td>Ascend-Receive-Secret</td>
</tr>
<tr>
<td>216</td>
<td>Ascend-IPX-Peer-Mode</td>
</tr>
<tr>
<td>217</td>
<td>Ascend-IP-Pool-Definition</td>
</tr>
<tr>
<td>218</td>
<td>Ascend-Assign-IP-Pool</td>
</tr>
<tr>
<td>219</td>
<td>Ascend-FR-Direct</td>
</tr>
<tr>
<td>220</td>
<td>Ascend-FR-Direct-Profile</td>
</tr>
<tr>
<td>221</td>
<td>Ascend-FR-Direct-DLCI</td>
</tr>
<tr>
<td>222</td>
<td>Ascend-Handle-IPX</td>
</tr>
<tr>
<td>212</td>
<td>Ascend-Netware-timeout</td>
</tr>
<tr>
<td>224</td>
<td>Ascend-IPX-Alias</td>
</tr>
<tr>
<td>225</td>
<td>Ascend-Metric</td>
</tr>
<tr>
<td>226</td>
<td>Ascend-PRI-Number-Type</td>
</tr>
<tr>
<td>227</td>
<td>Ascend-Dial-Number</td>
</tr>
<tr>
<td>228</td>
<td>Ascend-Route-IP</td>
</tr>
<tr>
<td>229</td>
<td>Ascend-Route-IPX</td>
</tr>
<tr>
<td>120</td>
<td>Ascend-Bridge</td>
</tr>
<tr>
<td>121</td>
<td>Ascend-Send-Auth</td>
</tr>
<tr>
<td>122</td>
<td>Ascend-Send-Password</td>
</tr>
<tr>
<td>123</td>
<td>Ascend-Link-Compression</td>
</tr>
<tr>
<td>124</td>
<td>Ascend-Target-Util</td>
</tr>
<tr>
<td>125</td>
<td>Ascend-Maximum-Channels</td>
</tr>
<tr>
<td>126</td>
<td>Ascend-Inc-Channel-Count</td>
</tr>
<tr>
<td>127</td>
<td>Ascend-Dec-Channel-Count</td>
</tr>
<tr>
<td>128</td>
<td>Ascend-Seconds-Of-History</td>
</tr>
<tr>
<td>129</td>
<td>Ascend-History-Weigh-Type</td>
</tr>
<tr>
<td>240</td>
<td>Ascend-Add-Seconds</td>
</tr>
<tr>
<td>241</td>
<td>Ascend-Remove-Seconds</td>
</tr>
<tr>
<td>242</td>
<td>Ascend-Data-Filter</td>
</tr>
<tr>
<td>243</td>
<td>Ascend-Call-Filter</td>
</tr>
<tr>
<td>244</td>
<td>Ascend-Idle-Limit</td>
</tr>
<tr>
<td>245</td>
<td>Ascend-Preempt-Limit</td>
</tr>
<tr>
<td>246</td>
<td>Ascend-Callback</td>
</tr>
<tr>
<td>247</td>
<td>Ascend-Data-Svc</td>
</tr>
</tbody>
</table>
Vendor-Specific Attributes

This section lists all vendor-specific attributes (VSAs) supported by Prime Access Registrar.

3GPP VSAs

Table C-4 lists the 3GPP VSAs. The vendor ID for 3GPP VSAs is 10415.

Table C-4 3GPP VSAs

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3GPP-IMSI</td>
<td>String</td>
<td>0-15</td>
</tr>
<tr>
<td>2</td>
<td>3GPP-Charging-Id</td>
<td>UINT</td>
<td>0-65535</td>
</tr>
<tr>
<td>3</td>
<td>3GPP-PDPType</td>
<td>ENUM</td>
<td>0-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 = IPv4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 = PPP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 = IPv6</td>
</tr>
<tr>
<td>4</td>
<td>3GPP-OG-Address</td>
<td>IP Address</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3GPP-GPRS-QoS-Profile</td>
<td>String</td>
<td>0-31</td>
</tr>
<tr>
<td>6</td>
<td>3GPP-SGSN-Address</td>
<td>IP Address</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>3GPP-GGSN-Address</td>
<td>IP Address</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-4 3GPP VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>3GPP-IMSI-MCC-MNC</td>
<td>String</td>
<td>6-6</td>
</tr>
<tr>
<td>9</td>
<td>3GPP-GGSN-MCC-MNC</td>
<td>String</td>
<td>6-6</td>
</tr>
<tr>
<td>10</td>
<td>3GPP-NSAPI</td>
<td>String</td>
<td>1-1</td>
</tr>
<tr>
<td>11</td>
<td>3GPP-Session-Stop-Indicator</td>
<td>String</td>
<td>2-2</td>
</tr>
<tr>
<td>12</td>
<td>3GPP-Selection-Mode</td>
<td>String</td>
<td>1-1</td>
</tr>
<tr>
<td>13</td>
<td>3GPP-Charging-Characteristics</td>
<td>String</td>
<td>4-4</td>
</tr>
<tr>
<td>14</td>
<td>3GPP-CG-IPv6-Address</td>
<td>String</td>
<td>16-16</td>
</tr>
<tr>
<td>15</td>
<td>3GPP-SGSN-IPv6-Address</td>
<td>String</td>
<td>16-16</td>
</tr>
<tr>
<td>16</td>
<td>3GPP-GGSN-IPv6-Address</td>
<td>String</td>
<td>6-6</td>
</tr>
<tr>
<td>17</td>
<td>3GPP-IPv6-DNS-Servers</td>
<td>String</td>
<td>16-253</td>
</tr>
<tr>
<td>18</td>
<td>3GPP-SGSN-MCC-MNC</td>
<td>String</td>
<td>0-1</td>
</tr>
<tr>
<td>19</td>
<td>3GPP-Teardown-Indicator</td>
<td>UINT32</td>
<td>0-1</td>
</tr>
<tr>
<td>20</td>
<td>3GPP-IMEISV</td>
<td>String</td>
<td>16-16</td>
</tr>
<tr>
<td>21</td>
<td>3GPP-RAT-Type</td>
<td>String</td>
<td>1-1</td>
</tr>
<tr>
<td>22</td>
<td>3GPP-User-Location-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>12</td>
<td>3GPP-MS-Timezone</td>
<td>String</td>
<td>2-2</td>
</tr>
<tr>
<td>24</td>
<td>3GPP-Camel-Charging-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>25</td>
<td>3GPP-Packet-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>26</td>
<td>3GPP-Negotiated-DSCP</td>
<td>String</td>
<td>1-1</td>
</tr>
</tbody>
</table>
3GPP2 VSAs

Table C-5 lists the 3GPP2 VSAs. The vendor ID for 3GPP2 VSAs is 5535 with 8-bit VendorTypeSize.

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CDMA-IKE-Pre-Shared-Secret-Request</td>
<td>ENUM</td>
<td>1-2; 1 = The PDSN requests a pre-shared secret for IKE 2 = The PDSN does not request a pre-shared secret for IKE</td>
</tr>
<tr>
<td>2</td>
<td>CDMA-Security-Level</td>
<td>ENUM</td>
<td>1-4; 1 = IPSec for registration messages 2 = IPSec for tunnels 3= IPSec for tunnels and registration messages 4 = No IPSec security</td>
</tr>
<tr>
<td>3</td>
<td>CDMA-Pre-Shared-Secret</td>
<td>String</td>
<td>0-24</td>
</tr>
<tr>
<td>4</td>
<td>CDMA-Reverse-Tunnel-Spec</td>
<td>ENUM</td>
<td>0-1; 0 = Reverse tunneling is not required 1 = Reverse tunneling is required</td>
</tr>
<tr>
<td>5</td>
<td>CDMA-Diff-Svc-Class-Opt</td>
<td>ENUM</td>
<td>0-46; 0 = Best Effort 10 = AF11 12 = AF12 14 = AF13 18 = AF21 20 = AF22 22 = AF12 26 = AF31 28 = AF32 30 = AF33 34 = AF41 36 = AF42 38 = AF43 46 = EF</td>
</tr>
<tr>
<td>6</td>
<td>CDMA-Container</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>7</td>
<td>CDMA-HA-IP-Addr</td>
<td>IPADDR</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>CDMA-KeyID-Attribute</td>
<td>String</td>
<td>0-28</td>
</tr>
<tr>
<td>9</td>
<td>CDMA-PCF-IP-Addr</td>
<td>IP Addr</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>CDMA-BS-MSC-Addr</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
### 3GPP2 VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>CDMA-User-ID</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>12</td>
<td>CDMA-Forward-MUX</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>13</td>
<td>CDMA-Reverse-MUX</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>14</td>
<td>CDMA-Forward-Rate</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>15</td>
<td>CDMA-Reverse-Rate</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>16</td>
<td>CDMA-Service-Option</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>17</td>
<td>CDMA-Forward-Type</td>
<td>ENUM 0-1;</td>
<td>0 = Primary</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 = Secondary</td>
</tr>
<tr>
<td>18</td>
<td>CDMA-Reverse-Type</td>
<td>ENUM 0-1;</td>
<td>0 = Primary</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 = Secondary</td>
</tr>
<tr>
<td>19</td>
<td>CDMA-Frame-Size</td>
<td>ENUM 0-2;</td>
<td>0 = No Fundamental</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 = 5 ms Frame and 20ms Mixed Frame</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 = 20 ms Frame</td>
</tr>
<tr>
<td>20</td>
<td>CDMA-Forward-RC</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>21</td>
<td>CDMA-Reverse-RC</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>22</td>
<td>CDMA-IP-Technology</td>
<td>ENUM 1-3;</td>
<td>1 = Simple-IP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 = Mobile-IP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 = Proxy-Mobile-IP</td>
</tr>
<tr>
<td>12</td>
<td>CDMA-Comp-Flag</td>
<td>ENUM 0-2;</td>
<td>0 = None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 = Non-secure</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 = Secure</td>
</tr>
</tbody>
</table>
### Table C-5  3GPP2 VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>CDMA-Release-Ind</td>
<td>ENUM</td>
<td>0-14; 0 = Unknown 1 = PPP/Service timeout 2 = Handoff 3 = PPP termination 4 = Mobile IP registration failure 5 = Abnormal Terminations 6 = Termination due to Resource management 7 = Service instance released 8 = Volume Quota reached, service instance released 9 = Duration Quota reached, Service instance released 10 = Incompatible PrePaid accounting information 11 = Airlink Parameter Change 12 = Time of Day Timer expiration 13 = Dormant by Accounting-Stop-triggered-by-Active-Stop 14 = Hot-Line status changed</td>
</tr>
<tr>
<td>25</td>
<td>CDMA-Dropped-Octets</td>
<td>UINT3</td>
<td>0-0</td>
</tr>
<tr>
<td>26</td>
<td>CDMA-Start-Date</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>27</td>
<td>CDMA-Start-Time</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>28</td>
<td>CDMA-Stop-Date</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>29</td>
<td>CDMA-Stop-Time</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>30</td>
<td>CDMA-Num-Active</td>
<td>UINT3</td>
<td>0-0</td>
</tr>
<tr>
<td>31</td>
<td>CDMA-SDB-Input-Octets</td>
<td>UINT3</td>
<td>0-0</td>
</tr>
<tr>
<td>32</td>
<td>CDMA-SDB-Output-Octets</td>
<td>UINT3</td>
<td>0-0</td>
</tr>
<tr>
<td>33</td>
<td>CDMA-NumSDB-Input</td>
<td>UINT3</td>
<td>0-0</td>
</tr>
<tr>
<td>34</td>
<td>CDMA-NumSDB-Output</td>
<td>UINT3</td>
<td>0-0</td>
</tr>
<tr>
<td>35</td>
<td>CDMA-Alt-Billing</td>
<td>UINT3</td>
<td>0-0</td>
</tr>
</tbody>
</table>
### Table C-5  3GPP2 VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>36</td>
<td>CDMA-IP-QoS</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>37</td>
<td>CDMA-Interconnect-IP</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>38</td>
<td>CDMA-Interconnect-QoS</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>39</td>
<td>CDMA-Air-QoS</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>40</td>
<td>CDMA-Airlink-Record-Type</td>
<td>ENUM 1-4</td>
<td>1 = Connection Setup 2 = Active Start 3 = Active Stop 4 = SDB Record</td>
</tr>
<tr>
<td>41</td>
<td>CDMA-R-P-Link-ID</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>42</td>
<td>CDMA-Airlink-Record-Type</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>43</td>
<td>CDMA-PPP-Bytes-Received</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>44</td>
<td>CDMA-Correlation-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>45</td>
<td>CDMA-Mobile-Terminate-Originated-Ind</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>46</td>
<td>CDMA-Inbound-Mobile-IP-Signalling-Octets</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>47</td>
<td>CDMA-Outbound-Mobile-IP-Signalling-Octets</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>48</td>
<td>CDMA-Session-Continue</td>
<td>ENUM 0-1</td>
<td>0 = False 1 = True</td>
</tr>
<tr>
<td>49</td>
<td>CDMA-Active-Time</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>50</td>
<td>CDMA-DCCH-Frame-Format</td>
<td>UINT3 2</td>
<td>0-3</td>
</tr>
<tr>
<td>51</td>
<td>CDMA-Beginning-Session</td>
<td>ENUM 0-1</td>
<td>0 = False 1 = True</td>
</tr>
<tr>
<td>52</td>
<td>CDMA-ESN</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>54</td>
<td>CDMA-S-Attribute</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>55</td>
<td>CDMA-S-Request-Attribute</td>
<td>ENUM 0-1</td>
<td>0 = The HA does not request a S secret for IKE 1 = The HA requests a S secret for IKE</td>
</tr>
</tbody>
</table>
### Appendix C  \hspace{4cm} \textbf{RADIUS Attributes}

### Vendor-Specific Attributes

#### Table C-5  \hspace{1cm} 3GPP2 VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>56</td>
<td>CDMA-S-Lifetime-Attribute</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>57</td>
<td>CDMA-MN-HA-SPI</td>
<td>String</td>
<td>0-4</td>
</tr>
<tr>
<td>58</td>
<td>CDMA-MN-HA-Shared-Key</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>59</td>
<td>CDMA-Remote-IPv4-Address</td>
<td>String</td>
<td>12-253</td>
</tr>
<tr>
<td>60</td>
<td>CDMA-HRPD-Access-Authentication</td>
<td>ENUM</td>
<td>1-1; 1 = HRPD Access Authentication</td>
</tr>
<tr>
<td>70</td>
<td>CDMA-Remote-IPv6-Address</td>
<td>String</td>
<td>68-253</td>
</tr>
<tr>
<td>71</td>
<td>CDMA-Remote-Address-Table-Index</td>
<td>UINT3 2</td>
<td>0-253</td>
</tr>
<tr>
<td>72</td>
<td>CDMA-Remote-IPv4-Address-Octet-Count</td>
<td>String</td>
<td>24-253</td>
</tr>
<tr>
<td>73</td>
<td>CDMA-Allowed-Differentiated-Service-Marking</td>
<td>String</td>
<td>12-253</td>
</tr>
<tr>
<td>74</td>
<td>CDMA-Service-Option-Profile</td>
<td>String</td>
<td>8-253</td>
</tr>
<tr>
<td>75</td>
<td>CDMA-DNS-Update-Required</td>
<td>ENUM</td>
<td>0-1; 0 = HA does not need to send DNS Update 1 = HA does need to send DNS Update</td>
</tr>
<tr>
<td>78</td>
<td>CDMA-Always-On</td>
<td>ENUM</td>
<td>0-1; 0 = Inactive 1 = Active</td>
</tr>
<tr>
<td>79</td>
<td>CDMA-Foreign-Agent-Address</td>
<td>IP Address</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>CDMA-Last-User-Activity</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>81</td>
<td>CDMA-MN-AAA-Removal-Indication</td>
<td>ENUM</td>
<td>1-1; 1 = MN-AAA not required</td>
</tr>
<tr>
<td>82</td>
<td>CDMA-RN-Packet-Data-Inactivity-Timer</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>83</td>
<td>CDMA-Forward-PDCH-RC</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>84</td>
<td>CDMA-Forward-DCCH-Mux-Option</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>85</td>
<td>CDMA-Reverse-DCCH-Mux-Option</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>86</td>
<td>CDMA-Forward-DCCH-RC</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>87</td>
<td>CDMA-Reverse-DCCH-RC</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
</tbody>
</table>
### Table C-5  3GPP2 VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>88</td>
<td>CDMA-Session-Termination-Capability</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>89</td>
<td>CDMA-Allowed-Persistent-TFTs</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>90</td>
<td>CDMA-PrePaid-Accounting-Quota</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>91</td>
<td>CDMA-PrePaid-Accounting-Capability</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>92</td>
<td>CDMA-MIP-Lifetime</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>93</td>
<td>CDMA-Accounting-Stop-Triggered-By-Active-Stop-Indication</td>
<td>ENUM 1-1; 1 = Accounting report at active/dormant transitions</td>
<td></td>
</tr>
<tr>
<td>94</td>
<td>CDMA-Service-Reference-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>95</td>
<td>CDMA-DNS-Update-Capability</td>
<td>ENUM 1-1; 1 = HA is capable of dynamic DNS Update</td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>CDMA-Disconnect-Reason</td>
<td>ENUM 1-1; 1 = MS Mobility Detection</td>
<td></td>
</tr>
<tr>
<td>97</td>
<td>CDMA-Remote-IPv6-Address-Octet-Count</td>
<td>String</td>
<td>36-253</td>
</tr>
<tr>
<td>98</td>
<td>CDMA-PrePaid-Tariff-Switching</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>99</td>
<td>CDMA-Authorization-Parameters</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>100</td>
<td>CDMA-BCMCS-Flow-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>101</td>
<td>CDMA-BCMCS-Capability</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>102</td>
<td>CDMA-Common-Session-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>103</td>
<td>CDMA-BSN-Session-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>104</td>
<td>CDMA-RN-Session-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>105</td>
<td>CDMA-Reason-Code</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>106</td>
<td>CDMA-Physical-Channel</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>107</td>
<td>CDMA-BCMCS-Flow-Transmission-Time</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>108</td>
<td>CDMA-Subnet</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>109</td>
<td>CDMA-Multicast-IP-Address</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>110</td>
<td>CDMA-Port</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>111</td>
<td>CDMA-Auth-Key</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>112</td>
<td>CDMA-TK-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>113</td>
<td>CDMA-BAK-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>114</td>
<td>CDMA-Reverse-PDCH-RC</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
<tr>
<td>115</td>
<td>CDMA-Acq-Info-Timestamp</td>
<td>UINT3 2</td>
<td>0-0</td>
</tr>
</tbody>
</table>
### Table C-10 3GPP2 VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>116</td>
<td>CDMA-MEID</td>
<td>String</td>
<td>0-16</td>
</tr>
<tr>
<td>117</td>
<td>CDMA-DNS-Server-IP-Address</td>
<td>String</td>
<td>0-22</td>
</tr>
<tr>
<td>118</td>
<td>CDMA-MIP6-Home-Agent-from-BU</td>
<td>String</td>
<td>0-18</td>
</tr>
<tr>
<td>119</td>
<td>CDMA-MIP6-CoA</td>
<td>String</td>
<td>0-22</td>
</tr>
<tr>
<td>120</td>
<td>CDMA-MIP6-HoA-Not-Authorized</td>
<td>ENUM</td>
<td>1-1; 1 = The HoA is not authorized</td>
</tr>
<tr>
<td>121</td>
<td>CDMA-MIP6-Session-Key</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>122</td>
<td>CDMA-Hot-Line-Accounting-Indication</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>112</td>
<td>CDMA-Hot-Line-Profile-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>124</td>
<td>CDMA-Filter-Rule</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>125</td>
<td>CDMA-HTTP-Redirection-Rule</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>126</td>
<td>CDMA-IP-Redirection-Rule</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>127</td>
<td>CDMA-Hot-Line-Capability</td>
<td>UINT3</td>
<td>0-0</td>
</tr>
<tr>
<td>128</td>
<td>CDMA-MIP6-Home-Link-Prefix</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>129</td>
<td>CDMA-MIP6-Home-Address</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>130</td>
<td>CDMA-Maximum-authorized-aggregate-bandwidth-for-best-effort-traffic</td>
<td>UINT3</td>
<td>0-0</td>
</tr>
<tr>
<td>131</td>
<td>CDMA-Authorized-QoS-Profile-IDs-for-the-User</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>132</td>
<td>CDMA-Granted-QoS-Parameters</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>133</td>
<td>CDMA-Maximum-Per-Flow-Priority-for-the-User</td>
<td>UINT3</td>
<td>0-15</td>
</tr>
<tr>
<td>134</td>
<td>CDMA-MIP6-Authenticator</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>135</td>
<td>CDMA-Source-IPv6-Address</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>136</td>
<td>CDMA-Program-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>137</td>
<td>CDMA-Program-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>138</td>
<td>CDMA-MIP6-MAC-Mobility-Data</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>139</td>
<td>CDMA-Inter-User-Priority</td>
<td>UINT3</td>
<td>0-3</td>
</tr>
<tr>
<td>140</td>
<td>CDMA-MIP6-Home-Agent-Attribute-B</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>141</td>
<td>CDMA-MIP6-HoA</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>142</td>
<td>CDMA-Carrier-ID</td>
<td>String</td>
<td>0-8</td>
</tr>
<tr>
<td>143</td>
<td>CDMA-GMT-Time-Zone-Offset</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
Table C-6 lists the ACC VSAs. The vendor ID for ACC VSAs is 5.

**Table C-6 ACC VSAs**

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Acc-Reason-Code</td>
<td>ENUM: no reason given/no failure resource shortage protocol error invalid attribute invalid service type invalid framed protocol invalid attribute value invalid user information invalid IP address invalid integer syntax invalid NAS port</td>
<td>0-56</td>
</tr>
<tr>
<td>1</td>
<td>Acc-Reason-Code</td>
<td>requested by user session already open network disconnect service interruption physical port error idle timeout session timeout administrative reset NAS reload or reset NAS error NAS request</td>
<td>0-56</td>
</tr>
<tr>
<td>1</td>
<td>Acc-Reason-Code</td>
<td>ENUM: undefined reason given too many RADIUS users conflicting attributes port limit exceeded facility not available internal configuration error bad route specification</td>
<td>0-56</td>
</tr>
</tbody>
</table>
### ACC VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Acc-Reason-Code (Continued)</td>
<td>Access Partition bind failure security violation request type conflict configuration disallowed missing attribute no authentication server invalid request missing parameter invalid parameter call cleared with cause inopportune config request invalid config parameter missing config parameter incompatible service profile administrative reset</td>
<td>0-56</td>
</tr>
<tr>
<td>1</td>
<td>Acc-Reason-Code (Continued)</td>
<td>administrative reload no authentication response port unneeded port preempted port suspended service unavailable callback user error host request no accounting server no accounting response access denied temporary buffer shortage</td>
<td>0-56</td>
</tr>
<tr>
<td>2</td>
<td>Acc-Ccp-Option</td>
<td>ENUM: Disabled Enabled</td>
<td>1-2</td>
</tr>
<tr>
<td>3</td>
<td>Acc-Input-Errors</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>4</td>
<td>Acc-Output-Errors</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>5</td>
<td>Acc-Access-Partition</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>6</td>
<td>Acc-Customer-Id</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>7</td>
<td>Acc-Ip-Gateway-Pri</td>
<td>IPADDR</td>
<td>0-253</td>
</tr>
<tr>
<td>8</td>
<td>Acc-Ip-Gateway-Sec</td>
<td>IPADDR</td>
<td>0-253</td>
</tr>
<tr>
<td>9</td>
<td>Acc-Route-Policy</td>
<td>ENUM: Funnel Direct</td>
<td>1-2</td>
</tr>
<tr>
<td>10</td>
<td>Acc-ML-MLX-Admin-State</td>
<td>ENUM: Enabled Disabled</td>
<td>1-2</td>
</tr>
<tr>
<td>11</td>
<td>Acc-ML-Call-Threshold</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
</tbody>
</table>
**Vendor-Specific Attributes**

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>Acc-ML-Clear-Threshold</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>13</td>
<td>Acc-ML-Damping-Factor</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>14</td>
<td>Acc-Tunnel-Secret</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>15</td>
<td>Acc-Clearing-Cause</td>
<td>ENUM:</td>
<td>0-127</td>
</tr>
<tr>
<td></td>
<td></td>
<td>cause unspecified</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>unassigned number</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>invalid information element</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>message incompatible with sta</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>recovery on timer expiration</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>mandatory information element</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>protocol error</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>interworking</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>normal clearing</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>user busy</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>no user responding</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>user alerted no answer</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Acc-Clearing-Cause</td>
<td>ENUM:</td>
<td>0-127</td>
</tr>
<tr>
<td></td>
<td>(Continued)</td>
<td>no route to transit network</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>call rejected</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>number changed</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>non selected user clearing</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>destination out of order</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>invalid or incomplete number</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>facility rejected</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>no route to destination</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>response to status inquiry</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>normal unspecified cause</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>no circuit or channel available</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>network out of order</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Acc-Clearing-Cause</td>
<td>ENUM:</td>
<td>0-127</td>
</tr>
<tr>
<td></td>
<td>(Continued)</td>
<td>temporary failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>switching equipment congestion</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>access information discarded</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>circuit or channel unavailable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>circuit or channel preempted</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>resources unavailable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>quality of service unavailable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>facility not subscribed</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>outgoing calls barred</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>incoming calls barred</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bearer capability unauthorized</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bearer capability not available</td>
<td></td>
</tr>
</tbody>
</table>

---

**Table C-6 ACC VSAs (continued)**
### Table C-6  ACC VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>Acc-Clearing-Cause <em>(Continued)</em></td>
<td>ENUM: channel unacceptable service not available bearer capability not implemented channel type not implemented facility not implemented call awarded being delivered restricted digital information service not implemented invalid call reference identified channel does not exist call identity does not exist call identity in use no call suspended</td>
<td>0-127</td>
</tr>
<tr>
<td>15</td>
<td>Acc-Clearing-Cause <em>(Continued)</em></td>
<td>ENUM: suspended call cleared incompatible destination invalid transit network selection invalid message mandatory information element message not implemented inopportune message information element not implemented</td>
<td>0-127</td>
</tr>
<tr>
<td>16</td>
<td>Acc-Clearing-Location</td>
<td>ENUM: local or remote user private network serving local beyond interworking point public network serving local transit network private network serving remote public network serving remote international network</td>
<td>0-10</td>
</tr>
<tr>
<td>17</td>
<td>Acc-Service-Profile</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>18</td>
<td>Acc-Request-Type</td>
<td>ENUM: Ring Indication Dial Request User Authentication Tunnel Authentication User Accounting Tunnel Accounting</td>
<td>1-6</td>
</tr>
<tr>
<td>19</td>
<td>Acc-Framed-Bridge</td>
<td>ENUM: Disabled Enabled</td>
<td>0-1</td>
</tr>
<tr>
<td>20</td>
<td>Acc-Vpsm-Oversubscribed</td>
<td>ENUM: False True</td>
<td>1-2</td>
</tr>
<tr>
<td>SubAttr</td>
<td>VSA Name</td>
<td>Type</td>
<td>Min-Max Value</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------</td>
<td>-------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>21</td>
<td>Acc-Acct-On-Off-Reason</td>
<td>ENUM:</td>
<td>0-5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NAS Reset</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NAS Reload</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Configuration Reset</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Configuration Reload</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enabled</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disabled</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>Acc-Tunnel-Port</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>12</td>
<td>Acc-Dns-Server-Pri</td>
<td>IPADDR</td>
<td>0-253</td>
</tr>
<tr>
<td>24</td>
<td>Acc-Dns-Server-Sec</td>
<td>IPADDR</td>
<td>0-253</td>
</tr>
<tr>
<td>26</td>
<td>Acc-Nbns-Server-Sec</td>
<td>IPADDR</td>
<td>0-253</td>
</tr>
<tr>
<td>27</td>
<td>Acc-Dial-Port-Index</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>Acc-Ip-Compression</td>
<td>ENUM:</td>
<td>0-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disabled</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enabled</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>Acc-Ipx-Compression</td>
<td>ENUM:</td>
<td>0-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Disabled</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Enabled</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>Acc-Connect-Tx-Speed</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>31</td>
<td>Acc-Connect-Rx-Speed</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>32</td>
<td>Acc-Modem-Modulation-Type</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>33</td>
<td>Acc-Modem-Error-Protocol</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>34</td>
<td>Acc-Callback-Delay</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>35</td>
<td>Acc-Callback-Num-Valid</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>36</td>
<td>Acc-Callback-Mode</td>
<td>ENUM:</td>
<td>0-7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>User-Auth</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>User-Specified-E-164</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CBCP-Callback</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLI-Callback</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>Acc-Callback-CBCP-Type</td>
<td>ENUM:</td>
<td>1-3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CBCP-None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CBCP-User-Specified</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CBCP-Pre-Specified</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>Acc-Dialout-Auth-Mode</td>
<td>ENUM:</td>
<td>1-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PAP</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CHAP</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CHAP-PAP</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NONE</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>Acc-Dialout-Auth-Password</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>40</td>
<td>Acc-Dialout-Auth-UserName</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
Table C-6  ACC VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>42</td>
<td>Acc-Access-Community</td>
<td>ENUM: PUBLIC, NETMAN</td>
<td>1-2</td>
</tr>
<tr>
<td>44</td>
<td>Acc-Ace-Token</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>45</td>
<td>Acc-Ace-Token-Ttl</td>
<td>UINT</td>
<td>0-253</td>
</tr>
<tr>
<td>46</td>
<td>Acc-Ip-Pool-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>47</td>
<td>Acc-Igmp-Admin-State</td>
<td>ENUM: Enabled, Disabled</td>
<td>1-2</td>
</tr>
<tr>
<td>48</td>
<td>Acc-Igmp-Version</td>
<td>ENUM: V1, V2</td>
<td>1-2</td>
</tr>
</tbody>
</table>

Altiga VSAs

Table C-7 lists the Altiga VSAs. The vendor ID for Altiga VSAs is 3076.

Table C-7  Altiga VSAs

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Altiga-General-Access-Hours</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>2</td>
<td>Altiga-General-Simultaneous-Logic</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>3</td>
<td>Altiga-General-Minimum-Password-Length</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>4</td>
<td>Altiga-General-All-Alphabetic-Only-Passwords</td>
<td>ENUM</td>
<td>0-1</td>
</tr>
<tr>
<td>5</td>
<td>Altiga-General-Primary-DNS</td>
<td>IP Address</td>
<td>0-253</td>
</tr>
<tr>
<td>6</td>
<td>Altiga-General-Secondary-DNS</td>
<td>IP Address</td>
<td>0-253</td>
</tr>
<tr>
<td>8</td>
<td>Altiga-General-Secondary-WINS</td>
<td>IP Address</td>
<td>0-253</td>
</tr>
<tr>
<td>9</td>
<td>Altiga-General-SEP-Card-Assignment</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>10</td>
<td>Altiga-General-Priority-On-SEP</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>11</td>
<td>Altiga-General-Tunneling-Protocol</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>12</td>
<td>Altiga-IPSec-Security-Associatio</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
### Table C-7  Altiga VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>Altiga-IPSec-Authentication</td>
<td>ENUM: None RADIUS LDAP NT Domain SDI Internal</td>
<td>0-5</td>
</tr>
<tr>
<td>15</td>
<td>Altiga-IPSec-Banner</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>16</td>
<td>Altiga-IPSec-Allow-Password-Storage-On-Client</td>
<td>ENUM: False True</td>
<td>0-1</td>
</tr>
<tr>
<td>17</td>
<td>Altiga-PPTP-L2TP-Use-Client-Specified-Addresses</td>
<td>ENUM: False True</td>
<td>0-1</td>
</tr>
<tr>
<td>18</td>
<td>Altiga-PPTP-Minimal-Authentication-Protocol</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>19</td>
<td>Altiga-L2TP-Minimal-Authentication</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>20</td>
<td>Altiga-PPTP-Encryption</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>21</td>
<td>Altiga-L2TP-Encryption</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>22</td>
<td>Altiga-Argument-Authentication-Server-Type</td>
<td>ENUM: First Active Server RADIUS LDAP NT SDI Internal</td>
<td>0-5</td>
</tr>
<tr>
<td>12</td>
<td>Altiga-Argument-Authentication-Server-Password</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>24</td>
<td>Altiga-Argument-Request-Authentication-Vector</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>25</td>
<td>Altiga-IPSec-LTL-Keepalives</td>
<td>ENUM: False True</td>
<td>0-1</td>
</tr>
<tr>
<td>26</td>
<td>Altiga-Argument-IPSec-Group-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>27</td>
<td>Altiga-IPSec-Split-Tunneling</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>28</td>
<td>Altiga-IPSec-Default-Domain</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>28</td>
<td>Altiga-IPSec-Secondary-Domain-List</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>30</td>
<td>Altiga-IPSec-Tunnel-Type</td>
<td>ENUM: LAN to LAN Remote Access</td>
<td>1-2</td>
</tr>
</tbody>
</table>
### Table C-7  Altiga VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>Altiga-IPSec-Mode-Configuration</td>
<td>ENUM: False True</td>
<td>0-1</td>
</tr>
<tr>
<td>32</td>
<td>Altiga-Argument-Authentication-Server-Priority</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>33</td>
<td>Altiga-IPSec-Group-Lock-Of-User</td>
<td>ENUM: False True</td>
<td>0-1</td>
</tr>
<tr>
<td>34</td>
<td>Altiga-IPSec-IPSec-Over-UDP</td>
<td>ENUM: False True</td>
<td>0-1</td>
</tr>
<tr>
<td>35</td>
<td>Altiga-IPSec-UDP-Port-For-IPSec</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>128</td>
<td>Altiga-Partitioning-Primary-DHCP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>129</td>
<td>Altiga-Partitioning-Secondary-DHCP</td>
<td>IP Address</td>
<td>0-253</td>
</tr>
<tr>
<td>131</td>
<td>Altiga-Partitioning-Premise-Route</td>
<td>IP Address</td>
<td>0-253</td>
</tr>
<tr>
<td>132</td>
<td>Altiga-Partitioning-Partition-Max-Sessions</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>133</td>
<td>Altiga-Partitioning-Mobile-IP-Key</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>134</td>
<td>Altiga-Partitioning-Mobile-IP-Address</td>
<td>IP Address</td>
<td>0-253</td>
</tr>
<tr>
<td>135</td>
<td>Altiga-Partitioning-Mobile-IP-SPI</td>
<td>IP Address</td>
<td>0-253</td>
</tr>
<tr>
<td>136</td>
<td>Altiga-Partitioning-Strip-Realm</td>
<td>ENUM: False True</td>
<td>0-1</td>
</tr>
<tr>
<td>137</td>
<td>Altiga-Partitioning-Group-ID</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>250</td>
<td>Altiga-Group-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
### Ascend VSAs

Table C-8 lists the Ascend VSAs. The vendor ID for Ascend VSAs is 529.

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>Ascend-Change-Password</td>
<td>String</td>
<td>0 - 253</td>
</tr>
<tr>
<td>18</td>
<td>Ascend-Session-Type</td>
<td>ENUM: Unused</td>
<td>0 - 8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unknown</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>G711-Ulaw</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>G711-Alaw</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>G712</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>G729</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>G712-64KPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>G728</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RT24</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Ascend-H312-Gatekeeper</td>
<td>IP Address</td>
<td>0 - 253</td>
</tr>
<tr>
<td>21</td>
<td>Ascend-H312-Conference-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>22</td>
<td>Ascend-H312-Destination-NAS-ID</td>
<td>IP Address</td>
<td>0-65535</td>
</tr>
<tr>
<td>12</td>
<td>Ascend-H312-Dialed-Time</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>24</td>
<td>Ascend-H312-Dialed-Number</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>25</td>
<td>Ascend-Inter-Arrival-Jitter</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>26</td>
<td>Ascend-Dropped-Octets</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>27</td>
<td>Ascend-Dropped-Packets</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>48</td>
<td>Ascend-Call-Direction</td>
<td>ENUM: Incoming</td>
<td>0-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Outgoing</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-8  
**Ascend VSAs (continued)**

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>49</td>
<td>Ascend-Service-Type</td>
<td>ENUM</td>
<td>0 - 12; NotUsed None EuUi Telnet TelnetBi n RawTcp TermSer ver MP VirtualC onn X25DChan PseuTun PPP IpFax Other ATM HdlcNrm VoIp Visa2 PPP Slip MPP X25 Combine t FR EuRaw</td>
</tr>
<tr>
<td>68</td>
<td>Ascend-Tunnel-ID</td>
<td>String</td>
<td>0 - 253</td>
</tr>
<tr>
<td>126</td>
<td>Ascend-Route-Preference</td>
<td>ENUM: Interface, OSPF-Internal , RIP, Down-WAN, OSPF-ASE, Infinite, ICMP</td>
<td>0-225</td>
</tr>
<tr>
<td>132</td>
<td>Ascend-Client-Gateway</td>
<td>IP Address</td>
<td>0 - 253</td>
</tr>
<tr>
<td>144</td>
<td>Ascend-Assign-IP-Client</td>
<td>IP Address</td>
<td>0-0</td>
</tr>
<tr>
<td>145</td>
<td>Ascend-Assign-IP-Server</td>
<td>IP Address</td>
<td>0-0</td>
</tr>
<tr>
<td>152</td>
<td>Ascend-Multicast-Rate-Limit</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>162</td>
<td>Ascend-FR-DCE-N392</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
</tbody>
</table>
### Table C-8  Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>163</td>
<td>Ascend-FR-DTE-N392</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>164</td>
<td>Ascend-FR-DCE-N393</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>165</td>
<td>Ascend-FR-DTE-N393</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>166</td>
<td>Ascend-FR-T391</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>167</td>
<td>Ascend-FR-T392</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>168</td>
<td>Ascend-Bridge-Address</td>
<td>UINT32</td>
<td>1-253</td>
</tr>
<tr>
<td>169</td>
<td>Ascend-TS-Idle-Limit</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>170</td>
<td>Ascend-TS-Idle-Mode</td>
<td>ENUM; TS-Idle-None, TS-Idle-Input, TS-Idle-Input-Output</td>
<td>0-2</td>
</tr>
<tr>
<td>171</td>
<td>Ascend-DBA-Monitor</td>
<td>ENUM; Transmit, Transmit-Receive, None</td>
<td>0-2</td>
</tr>
<tr>
<td>172</td>
<td>Ascend-Base-Channel-Count</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>173</td>
<td>Ascend-Minimum-Channels</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>174</td>
<td>Ascend-IPX-Route</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>175</td>
<td>Ascend-FT1-Caller</td>
<td>ENUM; FT1-No, FT1-Yes</td>
<td>0-1</td>
</tr>
<tr>
<td>176</td>
<td>Ascend-Backup</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>177</td>
<td>Ascend-Call-Type</td>
<td>ENUM; Nailed, Nailed/MPP, Perm/Switched</td>
<td>0-2</td>
</tr>
<tr>
<td>178</td>
<td>Ascend-Group</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>179</td>
<td>Ascend-FR-DLCI</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>180</td>
<td>Ascend-FR-Profile-Name</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>181</td>
<td>Ascend-Ara-PW</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>182</td>
<td>Ascend-IPX-Node-Address</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>183</td>
<td>Ascend-Home-Agent-IP-Addr</td>
<td>IP Address</td>
<td>0-0</td>
</tr>
<tr>
<td>184</td>
<td>Ascend-Home-Agent-Password</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>185</td>
<td>Ascend-Home-Network-Name</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>186</td>
<td>Ascend-Home-Agent-UDP-Port</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>187</td>
<td>Ascend-Multilink-ID</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
</tbody>
</table>
## Table C-8  Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>188</td>
<td>Ascend-Num-In-Multilink</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>189</td>
<td>Ascend-First-Dest</td>
<td>IP Address</td>
<td>0-0</td>
</tr>
<tr>
<td>190</td>
<td>Ascend-Pre-Input-Octets</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>191</td>
<td>Ascend-Pre-Output-Octets</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>192</td>
<td>Ascend-Pre-Input-Packets</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>193</td>
<td>Ascend-Pre-Output-Packets</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>194</td>
<td>Ascend-Maximum-Time</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>SubAttr</td>
<td>VSA Name</td>
<td>Type</td>
<td>Min-Max Value</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
</tbody>
</table>
### Table C-8 Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>195</td>
<td>vAscend-Pre-Output-Packets</td>
<td>ENUM: TermSrv-User-Quit, TermSrv-Idle-Timeout, TermSrv-Exit-Telnet, TermSrv-No-I Paddr, TermSrv-Exit-Raw-TCP, TermSrv-Exit-Login-Failed, TermSrv-Exit-Raw-TCP-Disabled, TermSrv-CTR L-C-In-Login, TermSrv-Destroyed, TermSrv-User-Closed-VCon, Call-Disconnected, TermSrv-VConn-Destroyed, TermSrv-Exit-Rlogin, TermSrv-Bad-Rlogin-Option, TermSrv-Not-Enough-Resources, MPP-No-NUL L-Msg-Timeout, CLID-Authentication-Failed, (continued)</td>
<td>0-195</td>
</tr>
</tbody>
</table>
### Table C-8  Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
</table>

(continued)
### Table C-8 Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>195</td>
<td>vAscend-Pre-Output-Packets <em>(continued)</em></td>
<td>Out-Of-Resources, Invalid-IP-Address, Hostname-Resolution-Failed, Bad-Or-Missing-Port-Number, Host-Reset, Connection-Refused, Connection-Timeout, Connection-Closed, Network-Unreachable, Host-Unreachable, Network-Unreachable-Admin, Host-Unreachable-Admin, Port-Unreachable,</td>
</tr>
</tbody>
</table>
# Table C-8 Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>196</td>
<td>Ascend-Connect-Progress</td>
<td>ENUM: No-Progress, unknown1, Call-Up, unknown2, Modem-Up, Modem-Awaiting-DCD, Modem-Awaiting-Codes, TermSrv-Started, TermSrv-Raw-TCP-Started, TermSrv-Telnet-Started, TermSrv-Raw-TCP-Connected, TermSrv-Tellnet-Connected, TermSrv-Rlogin-Started, TermSrv-Rlogin-Connected, TermSrv-Authentication-Begin, Modem-Outdial-Call-Up</td>
<td>0-94</td>
</tr>
</tbody>
</table>
### Table C-8  Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>196</td>
<td>Ascend-Connect-Progress</td>
<td>ENUM:</td>
<td>0-94</td>
</tr>
<tr>
<td>197</td>
<td>Ascend-Data-Rate</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
</tbody>
</table>
### Vendor-Specific Attributes

#### Table C-8 Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>198</td>
<td>Ascend-PreSession-Time</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>199</td>
<td>Ascend-Token-Idle</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>200</td>
<td>Ascend-Token-Idle</td>
<td>ENUM: Tok-Imm-No, Tok-Imm-Yes</td>
<td>0-1</td>
</tr>
<tr>
<td>210</td>
<td>Ascend-PPP-VJ-Slot-Comp</td>
<td>ENUM: VJ-Slot-Comp-No</td>
<td>1-1</td>
</tr>
<tr>
<td>211</td>
<td>Ascend-PPP-VJ-1172</td>
<td>ENUM: PPP-VJ-1172</td>
<td>1-1</td>
</tr>
<tr>
<td>212</td>
<td>Ascend-PPP-Async-Map</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>213</td>
<td>Ascend-Third-Prompt</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>214</td>
<td>Ascend-Send-Secret</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>215</td>
<td>Ascend-Receive-Secret</td>
<td>String</td>
<td>1-253</td>
</tr>
</tbody>
</table>
### Table C-8 Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>216</td>
<td>Ascend-IPX-Peer-Mode</td>
<td>ENUM:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>IPX-Peer-Router,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>IPX-Peer-Dialin</td>
<td></td>
</tr>
<tr>
<td>217</td>
<td>Ascend-IP-Pool-Definition</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>218</td>
<td>Ascend-Assign-IP-Pool</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>219</td>
<td>Ascend-FR-Direct</td>
<td>ENUM:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FR-Direct-No,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FR-Direct-Yes</td>
<td></td>
</tr>
<tr>
<td>220</td>
<td>Ascend-FR-Direct-Profile</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>221</td>
<td>Ascend-FR-Direct-DLCI</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>222</td>
<td>Ascend-Handle-IPX</td>
<td>ENUM:</td>
<td>0-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Handle-IPX-None,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Handle-IPX-Client,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Handle-IPX-Server</td>
<td></td>
</tr>
<tr>
<td>212</td>
<td>Ascend-Netware-timeout</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>224</td>
<td>Ascend-IPX-Alias</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>225</td>
<td>Ascend-Metric</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>226</td>
<td>Ascend-PRI-Number-Type</td>
<td>ENUM:</td>
<td>0-5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unknown-Number,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Intl-Number,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>National-Number,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Local-Number,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Abbrev-Number</td>
<td></td>
</tr>
<tr>
<td>227</td>
<td>Ascend-Dial-Number</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>228</td>
<td>Ascend-Route-IP</td>
<td>ENUM:</td>
<td>0-5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unknown-Number,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Intl-Number,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>National-Number,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Local-Number,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Abbrev-Number</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-8 Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>229</td>
<td>Ascend-Route-IPX</td>
<td>ENUM: Route-IPX-No Route-IPX-Yes</td>
<td>0-1</td>
</tr>
<tr>
<td>120</td>
<td>Ascend-Bridge</td>
<td>ENUM: Bridge-No, Bridge-Yes</td>
<td>0-1</td>
</tr>
<tr>
<td>121</td>
<td>Ascend-Send-Auth</td>
<td>ENUM: Send-Auth-No, Send-Auth-CH AP</td>
<td>0-2</td>
</tr>
<tr>
<td>122</td>
<td>Ascend-Send-Passwd</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>123</td>
<td>Ascend-Link-Compression</td>
<td>ENUM: Link-Comp-One, Link-Comp-St ac, Link-Comp-St ac-Draft-9, Link-Comp-M S-Stac</td>
<td>0-3</td>
</tr>
<tr>
<td>124</td>
<td>Ascend-Target-Util</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>125</td>
<td>Ascend-Maximum-Channels</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>126</td>
<td>Ascend-Inc-Channel-Count</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>127</td>
<td>Ascend-Dec-Channel-Count</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>128</td>
<td>Ascend-Seconds-Of-History</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>129</td>
<td>Ascend-History-Weigh-Type</td>
<td>ENUM: History-Const ant, History-Linear, History-Quadr atic</td>
<td>0-2</td>
</tr>
<tr>
<td>240</td>
<td>Ascend-Add-Seconds</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>241</td>
<td>Ascend-Remove-Seconds</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>242</td>
<td>Ascend-Data-Filter</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>243</td>
<td>Ascend-Call-Filter</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>244</td>
<td>Ascend-Idle-Limit</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>245</td>
<td>Ascend-Idle-Limit</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
</tbody>
</table>
### Table C-8  Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>246</td>
<td>Ascend-Callback</td>
<td>ENUM: Callback-No, Callback-Yes</td>
<td>0-1</td>
</tr>
</tbody>
</table>
## Vendor-Specific Attributes

### Table C-8  Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
</table>
Table C-8 Ascend VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>248</td>
<td>Ascend-Force-56</td>
<td>ENUM: Force-56-No, Force-56-Yes</td>
<td>0-1</td>
</tr>
<tr>
<td>249</td>
<td>Ascend-Billing-Number</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>250</td>
<td>Ascend-Call-By-Call</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>251</td>
<td>Ascend-Transit-Number</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>252</td>
<td>Ascend-Host-Info</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>253</td>
<td>Ascend-PPP-Address</td>
<td>IP Address</td>
<td>0-0</td>
</tr>
<tr>
<td>254</td>
<td>Ascend-MPP-Idle-Percent</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
</tbody>
</table>

Bay Networks VSAs

Table C-9 lists the Bay Networks VSAs. The vendor ID for Bay Networks VSAs is 1584.

Table C-9 Bay Networks VSAs

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>Annex-Filter</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>29</td>
<td>Annex-CLI-Command</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>30</td>
<td>Annex-CLI-Filter</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>31</td>
<td>Annex-Host-Restrict</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>32</td>
<td>Annex-Host-Allow</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>33</td>
<td>Annex-Product-Name</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>34</td>
<td>Annex-SW-Version</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>35</td>
<td>Annex-Local-IP-Address</td>
<td>IPADDR</td>
<td>1-253</td>
</tr>
<tr>
<td>36</td>
<td>Annex-Callback-Portlist</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>44</td>
<td>Annex-System-Disc-Reason</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>45</td>
<td>Annex-Modem-Disc-Reason</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>46</td>
<td>Annex-Disconnect-Reason</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>50</td>
<td>Annex-Transmit-Speed</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>51</td>
<td>Annex-Receive-Speed</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
</tbody>
</table>
Cabletron VSAs

Table C-10 lists the Cabletron VSAs. The vendor ID for Cabletron VSAs is 52.

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>192</td>
<td>Cabletron-Framed-Data-Rate</td>
<td>ENUM:</td>
<td>0-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rate-56KB</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rate-64KB</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rate-112KB</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rate-128KB</td>
<td></td>
</tr>
<tr>
<td>193</td>
<td>Cabletron-Phone-Number</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>194</td>
<td>Cabletron-Caller-Id</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>196</td>
<td>Cabletron-Connection-Reference</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>198</td>
<td>Cabletron-Initial-Rate</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>199</td>
<td>Cabletron-Maximum-Rate</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>192</td>
<td>Cabletron-Framed-Data-Rate</td>
<td>Enum:</td>
<td>192</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rate-56KB</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rate-64KB</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rate-112KB</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rate-128KB</td>
<td></td>
</tr>
</tbody>
</table>

Cisco Prime Access Registrar Internal VSAs

Table C-11 lists the Prime Access Registrar Internal VSAs. The vendor ID for Prime Access Registrar internal VSAs is 1760.

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Realm</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>2</td>
<td>Incoming-Translation-Groups</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>3</td>
<td>Client-IP-Address</td>
<td>IP Address</td>
<td>1-253</td>
</tr>
<tr>
<td>4</td>
<td>Subnet-Mask</td>
<td>IP Address</td>
<td>1-253</td>
</tr>
<tr>
<td>5</td>
<td>Outgoing-Translation-Groups</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>6</td>
<td>Authentication-Service</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>7</td>
<td>Authorization-Service</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>8</td>
<td>DNIS</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>9</td>
<td>CLID</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>10</td>
<td>UserFilterMask</td>
<td>String</td>
<td>1-253</td>
</tr>
</tbody>
</table>
### Table C-11  Prime Access Registrar Internal VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>Session-Manager</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>12</td>
<td>Accounting-Service</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>13</td>
<td>TimeRange</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>14</td>
<td>AcceptedProfiles</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>15</td>
<td>Policy</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>16</td>
<td>Prefix</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>17</td>
<td>Delimiters</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>18</td>
<td>StripPrefix</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>19</td>
<td>ODBC-Reply-Attrs</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>20</td>
<td>ODBC-Check-Attrs</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>21</td>
<td>Session-Service</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>22</td>
<td>Prepaid</td>
<td>ENUM:</td>
<td>0-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = False</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = True</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Suffix</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>12</td>
<td>Implicit-Auth-Enabled</td>
<td>ENUM:</td>
<td>0-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = False</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = True</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>StripSuffix</td>
<td>ENUM:</td>
<td>0-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = False</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = True</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>Query-Service</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>92</td>
<td>RepSourceIP</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>93</td>
<td>RepTargetIP</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>94</td>
<td>RepTxnNum</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>95</td>
<td>RepTxnCRC</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>96</td>
<td>RepTxnElementCount</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>97</td>
<td>RepNeedsFullSync</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>98</td>
<td>RepNeedsReSync</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>99</td>
<td>RepLastRxTxnNum</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>100</td>
<td>RepLastRxTxnCRC</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>101</td>
<td>RepNeedsMember</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>102</td>
<td>RepMemberName</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>103</td>
<td>RepMemberIP</td>
<td>IP Address</td>
<td>0-253</td>
</tr>
<tr>
<td>104</td>
<td>RepMemberPort</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>105</td>
<td>RepMemberOrdinal</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
</tbody>
</table>
Table C-11  Prime Access Registrar Internal VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>106</td>
<td>RepWorkLoad</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>107</td>
<td>RepTxTime</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>108</td>
<td>RepElementPath</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>109</td>
<td>RepElementValue</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>110</td>
<td>RepElementOrdinal</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>111</td>
<td>RepElementCRC</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>112</td>
<td>RepElementType</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>113</td>
<td>RepElementMode</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>114</td>
<td>RepPartialElement</td>
<td>Undefined</td>
<td>0-253</td>
</tr>
</tbody>
</table>

Cisco VSAs

Table C-12 lists the Cisco VSAs. The vendor ID for Cisco VSAs is 9.

Table C-12  Cisco VSAs

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cisco-AVPair</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>2</td>
<td>Cisco-NAS-Port</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>3</td>
<td>Cisco-Fax-Account-ID-Origin</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>4</td>
<td>Cisco-Fax-Message-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>5</td>
<td>Cisco-Fax-Pages</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>6</td>
<td>Cisco-FAX Cover Page Flag</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>7</td>
<td>Cisco-Fax-Modem-Time</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>8</td>
<td>Cisco-Fax-Connect-Speed</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>9</td>
<td>Cisco-Fax-Recipient-Count</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>10</td>
<td>Cisco-Fax-Process-Abort-Flag</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>11</td>
<td>Cisco-Fax-DSN-Address</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>12</td>
<td>Cisco-Fax-DSN-Flag</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>13</td>
<td>Cisco-Fax-MDN-Address</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>14</td>
<td>Cisco-Fax-MDN-Flag</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>15</td>
<td>Cisco-Fax-Auth-Status</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>16</td>
<td>Cisco-Email-Server-Address</td>
<td>IP Address</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-12  Cisco VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>Cisco-Email-Server-ACK Flag</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>18</td>
<td>Cisco-Gateway-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>19</td>
<td>Cisco-Call-Type</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>20</td>
<td>Cisco-Port-Used</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>21</td>
<td>Cisco-Abort-Cause</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>22</td>
<td>Cisco-CRS-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>12</td>
<td>Cisco-h312-Remote-Address</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>24</td>
<td>Cisco-h312-Conf-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>25</td>
<td>Cisco-h312-Setup-Time</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>26</td>
<td>Cisco-h312-Call-Origin</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>27</td>
<td>Cisco-h312-Call-Type</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>28</td>
<td>Cisco-h312-Connect-Time</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>29</td>
<td>Cisco-h312-Disconnect-Time</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>30</td>
<td>Cisco-h312-Disconnect-Cause</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>31</td>
<td>Cisco-h312-Voice-Quality</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>32</td>
<td>Cisco-h312-Generic-IVR-Out</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>33</td>
<td>Cisco-h312-Gateway-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>34</td>
<td>Cisco-3GPP2-AVPair</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>35</td>
<td>Cisco Connection ID-h312-incoming-connection-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>100</td>
<td>Cisco-h312-Generic-IVR-In</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>101</td>
<td>Cisco-h312-Amount-Balance</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>102</td>
<td>Cisco-h312-Time-Balance</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>103</td>
<td>Cisco-h312-Return-Code</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>104</td>
<td>Cisco-h312-Prompt-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>105</td>
<td>Cisco-h312-Time-of-Day</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>106</td>
<td>Cisco-h312-Redirect-Number</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>107</td>
<td>Cisco-h312-Preferred-Language</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>108</td>
<td>Cisco-h312-Redirect-IP-Address</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>109</td>
<td>Cisco-h312-Billing-Model</td>
<td>ENUM:</td>
<td>0-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>postpaid,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>prepaid</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>Cisco-h312-Currency</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
### Table C-12  Cisco VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>Cisco-UCP-IP-Pool-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>129</td>
<td>Cisco-UCP-User-Max-Sessions</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>130</td>
<td>Cisco-UCP-User-Session-Count</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>131</td>
<td>Cisco-UCP-Next-Session-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>132</td>
<td>Cisco-UCP-VPDN-Max-Sessions</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>133</td>
<td>Cisco-UCP-VPDN-Session-Count</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>134</td>
<td>Cisco-UCP-B-Channel-Max-Sessions</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>135</td>
<td>Cisco-UCP-B-Channel-Session-Count</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>136</td>
<td>Cisco-UCP-Status</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>137</td>
<td>Cisco-UCP-BLOB-Attribute-Length</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>138</td>
<td>Cisco-UCP-Disable-Status</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>139</td>
<td>Cisco-UCP-Block-Access-Range</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>140</td>
<td>Cisco-UCP-Home-POP-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>175</td>
<td>Cisco-UCP-IP-Addresses</td>
<td>IP Addresses</td>
<td>0-253</td>
</tr>
<tr>
<td>176</td>
<td>Cisco-UCP-Session-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>211</td>
<td>Cisco-Ascend AV pairs</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>250</td>
<td>Cisco-SSG-Account-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>251</td>
<td>Cisco-SSG-Service-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>252</td>
<td>Cisco-SSG-Command-Code</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>253</td>
<td>Cisco-SSG-Control-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
Compatible VSAs

Table C-13 lists the Compatible VSAs. The vendor ID for Compatible VSAs is 255.

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Compatible-Tunnel-Delay</td>
<td>UNIT32</td>
<td>0-253</td>
</tr>
<tr>
<td>1</td>
<td>Compatible-Tunnel-Throughput</td>
<td>UNIT32</td>
<td>0-253</td>
</tr>
<tr>
<td>3</td>
<td>Compatible-Tunnel-Server-Endpoint</td>
<td>IP Address</td>
<td>0-253</td>
</tr>
<tr>
<td>4</td>
<td>Compatible-Tunnel-Group-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>5</td>
<td>Compatible-Tunnel-Password</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>6</td>
<td>Compatible-Echo</td>
<td>UNIT32</td>
<td>0-253</td>
</tr>
<tr>
<td>7</td>
<td>Compatible-Tunnel-Client-IPX</td>
<td>UNIT32</td>
<td>0-253</td>
</tr>
</tbody>
</table>

Microsoft VSAs

Table C-14 lists the Microsoft VSAs. The vendor ID for Microsoft VSAs is 311.

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MS-CHAP-Response</td>
<td>String</td>
<td>50-50</td>
</tr>
<tr>
<td>2</td>
<td>MS-CHAP-Error</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>3</td>
<td>MS-CHAP-CPW1</td>
<td>String</td>
<td>70-70</td>
</tr>
<tr>
<td>4</td>
<td>MS-CHAP-CPW2</td>
<td>String</td>
<td>84-84</td>
</tr>
<tr>
<td>5</td>
<td>MS-CHAP-LM-Enc-PW</td>
<td>String</td>
<td>4-253</td>
</tr>
<tr>
<td>6</td>
<td>MS-CHAP-NT-Enc-PW</td>
<td>String</td>
<td>4-253</td>
</tr>
<tr>
<td>7</td>
<td>MS-MPPE-Encryption-Policy</td>
<td>ENUM: Encryption-Allowed Encryption-Required</td>
<td>1-2</td>
</tr>
<tr>
<td>8</td>
<td>MS-MPPE-Encryption-Types</td>
<td>String</td>
<td>0-4</td>
</tr>
<tr>
<td>9</td>
<td>MS-RAS-Vendor</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>10</td>
<td>MS-CHAP-Domain</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>11</td>
<td>MS-CHAP-Challenge</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>12</td>
<td>MS-CHAP-MPPE-Keys</td>
<td>String</td>
<td>32-32</td>
</tr>
<tr>
<td>13</td>
<td>MS-BAP-Usage</td>
<td>ENUM: Not allowed Allowed Required</td>
<td>0-2</td>
</tr>
</tbody>
</table>
### Table C-14  Microsoft VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>MS-Link-Utilization-Threshhold</td>
<td>UINT32 0-253</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>MS-Link-Drop-Time-Limit</td>
<td>String 0-253</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>MS-MPPE-Send-Key</td>
<td>String 0-253</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>MS-MPPE-Recv-Key</td>
<td>String 0-253</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>MS-RAS-Version</td>
<td>String 0-253</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>MS-Old-ARAP-Password</td>
<td>String 0-253</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>MS-New-ARAP-Password</td>
<td>String 0-253</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>MS-ARAP-Password-Change-Reason</td>
<td>ENUM: Just-Change-Passw</td>
<td>1-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ord-Expired-Passw</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ord-Admin-Requires-Pas</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>sword-Chang</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Password-Too-Short</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>MS-Filter</td>
<td>String 0-253</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>MS-Acct-Auth-Type</td>
<td>ENUM: PAP CHAP MS-CHA</td>
<td>1-5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>P-1 MS-CHAP-2 EAP</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>MS-CHAP2-Success</td>
<td>String 43-43</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>MS-CHAP2-CPW8</td>
<td>String 68-68</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>MS-Secondary-DNS-Server</td>
<td>IP Address 68-68</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>MS-Secondary-NBNS-Server</td>
<td>IP Address 70-70</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>MS-ARAP-Challenge</td>
<td>String 8-8</td>
<td></td>
</tr>
</tbody>
</table>
Nomadix VSAs

Table C-15 lists the Nomadix VSAs. The vendor ID for Nomadix VSAs is 3309.

Table C-15 Nomadix VSAs

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Nomadix-Bw-Up</td>
<td>0</td>
<td>253</td>
</tr>
<tr>
<td>2</td>
<td>Nomadix-Dw-Down</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
</tbody>
</table>

RedBack VSAs

Table C-16 lists the RedBack VSAs. The vendor ID for RedBack VSAs is 1252.

Table C-16 RedBack VSAs

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RedBack-Client-DNS-Pri</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>2</td>
<td>RedBack-Client-DNS-Sec</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>3</td>
<td>RedBack-DHCP-Max-Leases</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>4</td>
<td>RedBack-Context-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>5</td>
<td>RedBack-Bridge-Group</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>6</td>
<td>RedBack-BG-Aging-Time</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>7</td>
<td>RedBack-BG-Path-Cost</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>8</td>
<td>RedBack-BG-Span-Dis</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>9</td>
<td>RedBack-BG-Trans-BPDU</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>10</td>
<td>RedBack-Rate-Limit-Rate</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>11</td>
<td>RedBack-Rate-Limit-Burst</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>12</td>
<td>RedBack-Police-Rate</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>13</td>
<td>RedBack-Police-Burst</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>14</td>
<td>RedBack-Source-Validation</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>15</td>
<td>RedBack-Tunnel-Domain</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>16</td>
<td>RedBack-Tunnel-Local-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>17</td>
<td>RedBack-Tunnel-Remote-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>18</td>
<td>RedBack-Tunnel-Function</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>21</td>
<td>RedBack-Tunnel-Max-Sessions</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>22</td>
<td>RedBack-Tunnel-Max-Tunnels</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>12</td>
<td>RedBack-Tunnel-Session-Auth</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>24</td>
<td>RedBack-Tunnel-Window</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>25</td>
<td>RedBack-Tunnel-Retransmit</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
### Table C-16 RedBack VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>RedBack-Tunnel-Cmd-Timeout</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>27</td>
<td>RedBack-PPPOE-URL</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>28</td>
<td>RedBack-PPPOE-MOTM</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>29</td>
<td>RedBack-Tunnel-Group</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>30</td>
<td>RedBack-Tunnel-Context</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>31</td>
<td>RedBack-Tunnel-Algorithm</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>32</td>
<td>RedBack-Tunnel-Deadtime</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>33</td>
<td>RedBack-Mcast-Send</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>34</td>
<td>RedBack-Mcast-Receive</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>35</td>
<td>RedBack-Mcast-MaxGroups</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>36</td>
<td>RedBack-Ip-Address-Pool-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>37</td>
<td>RedBack-Tunnel-DNIS</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>38</td>
<td>RedBack-Medium-Type</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>39</td>
<td>RedBack-PVC-Encapsulation-Type</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>40</td>
<td>RedBack-PVC-Profile-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>41</td>
<td>RedBack-PVC-Circuit-Padding</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>42</td>
<td>RedBack-Bind-Type</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>43</td>
<td>RedBack-Bind-Auth-Protocol</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>44</td>
<td>RedBack-Bind-Auth-Max-Sessions</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>45</td>
<td>RedBack-Bind-Bypass-Bypass</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>46</td>
<td>RedBack-Bind-Auth-Context</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>47</td>
<td>RedBack-Bind-Auth-Service-Grp</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>48</td>
<td>RedBack-Bind-Bypass-Context</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>49</td>
<td>RedBack-Bind-Int-Context</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>50</td>
<td>RedBack-Bind-Tun-Context</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>51</td>
<td>RedBack-Bind-Ses-Context</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>52</td>
<td>RedBack-Bind-Dot1q-Slot</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>53</td>
<td>RedBack-Bind-Dot1q-Port</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>54</td>
<td>RedBack-Bind-Dot1q-Vlan-Tag-Id</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>55</td>
<td>RedBack-Bind-Int-Interface-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>56</td>
<td>RedBack-Bind-L2TP-Tunnel-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
### Table C-16  RedBack VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>57</td>
<td>RedBack-Bind-L2TP-Flow-Control</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>58</td>
<td>RedBack-Bind-Sub-User-At-Context</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>59</td>
<td>RedBack-Bind-Sub-Password</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>60</td>
<td>RedBack-Ip-Host-Addr</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>61</td>
<td>RedBack-IP-TOS-Field</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>62</td>
<td>RedBack-NAS-Real-Port</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>63</td>
<td>RedBack-Tunnel-Session-Auth-Ctx</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>64</td>
<td>RedBack-Tunnel-Session-Auth-Service-Grp</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>65</td>
<td>RedBack-Tunnel-Rate-Limit-Rate</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>66</td>
<td>RedBack-Tunnel-Rate-Limit-Burst</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>67</td>
<td>RedBack-Tunnel-Police-Rate</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>68</td>
<td>RedBack-Tunnel-Police-Burst</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>69</td>
<td>RedBack-Tunnel-L2F-Second-Password</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>128</td>
<td>RedBack-Acct-Input-Octets-64</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>129</td>
<td>RedBack-Acct-Output-Octets-64</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>130</td>
<td>RedBack-Acct-Input-Packets-64</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>131</td>
<td>RedBack-Acct-Output-Packets-64</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>132</td>
<td>RedBack-Assigned-IP-Address</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>133</td>
<td>RedBack-Acct-Mcast-In-Octets</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>134</td>
<td>RedBack-Acct-Mcast-In-Octets</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>135</td>
<td>RedBack-Acct-Mcast-In-Packets</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>136</td>
<td>RedBack-Acct-Mcast-Out-Packets</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>137</td>
<td>RedBack-LAC-Port</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>138</td>
<td>RedBack-LAC-Real-Port</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>139</td>
<td>RedBack-LAC-Port-Type</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>140</td>
<td>RedBack-LAC-Real-Port-Type</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
RedCreek VSAs

Table C-17 lists the RedCreek VSAs. The vendor ID for RedCreek VSAs is 1958.

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>RedCreek-Tunneled-IP-Netmask</td>
<td>IP Address</td>
<td>0-253</td>
</tr>
<tr>
<td>7</td>
<td>RedCreek-Tunneled-Gateway</td>
<td>IP Address</td>
<td>0-253</td>
</tr>
<tr>
<td>9</td>
<td>RedCreek-Tunneled-WINS-Server1</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>10</td>
<td>RedCreek-Tunneled-WINS-Server2</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>11</td>
<td>RedCreek-Tunneled-HostName</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>12</td>
<td>RedCreek-Tunneled-DomainName</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>13</td>
<td>RedCreek-Tunneled-Search-List</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>

TACACS+ VSAs

Table C-18 lists the TACACS+ VSAs. The vendor ID for TACACS+ VSAs is 268435456.

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tacacs-Version</td>
<td>ENUM:</td>
<td>0-255</td>
</tr>
<tr>
<td></td>
<td>192 = 12.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>193 = 12.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Tacacs-Type</td>
<td>ENUM:</td>
<td>1-3</td>
</tr>
<tr>
<td></td>
<td>1 = Authentication</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 = Authorization</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 = Accounting</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Tacacs-Sequence-Number</td>
<td>UINT32</td>
<td>0-1</td>
</tr>
<tr>
<td>4</td>
<td>Tacacs-Session-Id</td>
<td>UINT32</td>
<td>0-2147483647</td>
</tr>
<tr>
<td>5</td>
<td>Tacacs-Action</td>
<td>ENUM:</td>
<td>0-253</td>
</tr>
<tr>
<td></td>
<td>1 = Login</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 = ChPass</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 = SendPass</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 = SendAuth</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Tacacs-Privilege-Level</td>
<td>UINT32</td>
<td>0-15</td>
</tr>
<tr>
<td>7</td>
<td>Tacacs-Authentication-Type</td>
<td>ENUM:</td>
<td>1-5</td>
</tr>
<tr>
<td></td>
<td>1 = ASCII</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 = PAP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 = CHAP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 = ARAP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 = MSCHAP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table C-18  TACACS+ VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Tacacs-Service</td>
<td>ENUM:</td>
<td>1-9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = Login</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = Enable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 = PPP</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 = ARAP</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 = PT</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 = RCMD</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 = X25</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 = NASI</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>9 = FWPROXY</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Tacacs-User-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>10</td>
<td>Tacacs-Port</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>11</td>
<td>Tacacs-Remote-Address</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>12</td>
<td>Tacacs-Data</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>13</td>
<td>Tacacs-User-Message</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>14</td>
<td>Tacacs-User-Data</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>15</td>
<td>Tacacs-Authentication-Continue-Flag</td>
<td>ENUM:</td>
<td>0-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = Continue</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = Abort</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Tacacs-Authentication-Reply-Flag</td>
<td>ENUM:</td>
<td>0-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = Echo</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = NoEcho</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Tacacs-Authentication-Reply-Status</td>
<td>ENUM:</td>
<td>0-33</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = Pass</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = Fail</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 = GetData</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 = GetUser</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 = GetPass</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 = Restart</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 = Error</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>33 = Follow</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Tacacs-Authorization-Reply-Status</td>
<td>ENUM:</td>
<td>0-33</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = PassAdd</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = PassRepl</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>16 = Fail</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>17 = Error</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>33 = Follow</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Tacacs-Server-Message</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
### Table C-18  TACACS+ VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>Tacacs-Authentication-Method</td>
<td>ENUM:</td>
<td>0-32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = NotSet</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = KRB5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 = Line</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 = Enable</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 = Local</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 = TacacsPlus</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 = Guest</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>16 = Radius</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>17 = KRB4</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>32 = RCMD</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Tacacs-AVPair</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>22</td>
<td>Tacacs-Accounting-Reply-Status</td>
<td>ENUM:</td>
<td>0-33</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = Success</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = Fail</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>33 = Follow</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Tacacs-Header-Flag</td>
<td>ENUM:</td>
<td>0-5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = Encrypted</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = Unencrypted</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 = Encrypted + ReuseConnection</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 = Unencrypted + ReuseConnection</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>Tacacs-User-Password</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>25</td>
<td>Tacacs-Accounting-Request-Flag</td>
<td>ENUM:</td>
<td>0-33</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = More</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = Start</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 = Start</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 = Stop</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 = Stop</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 = Start</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 = Start</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 = Update</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>9 = More</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10 = Start</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>11 = Start</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>12 = Stop</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>13 = Stop</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>14 = Start</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>15 = Start</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>Tacacs-CHAP-Password</td>
<td>CHAP_PASSWORD</td>
<td>17-17</td>
</tr>
<tr>
<td>27</td>
<td>Tacacs-CHAP-Challenge</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
Appendix C      RADIUS Attributes

Table C-18      TACACS+ VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>Tacacs-MSCHAP-Response</td>
<td>String</td>
<td>50-50</td>
</tr>
<tr>
<td>29</td>
<td>Tacacs-MSCHAP-Challenge</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>

Teledit VSAs

Table C-19 lists the Telebit VSAs. The vendor ID for Telebit VSAs is 117.

Table C-19      Telebit VSAs

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Telebit-Login-Command</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>2</td>
<td>Telebit-Port-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>3</td>
<td>Telebit-activate-Command</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>4</td>
<td>Telebit-Accounting-Info</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>5</td>
<td>Telebit-Login-Option</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>

Unisphere VSAs

Table C-20 lists the Unisphere VSAs. The vendor ID for RedBack VSAs is 4874.

Table C-20      Unisphere VSAs

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unisphere-Virtual-Router</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>2</td>
<td>Unisphere-Local-Address-Pool</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>3</td>
<td>Unisphere-Local-Interface</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>4</td>
<td>Unisphere-Primary-DNS</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>5</td>
<td>Unisphere-Secondary-DNS</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>6</td>
<td>Unisphere-Primary-WINS</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>7</td>
<td>Unisphere-Secondary-WINS</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>8</td>
<td>Unisphere-Tunnel-Virtual-Router</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>9</td>
<td>Unisphere-Tunnel-Password</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>10</td>
<td>Unisphere-Ingress-Policy-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>11</td>
<td>Unisphere-Egress-Policy-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
### USR VSAs

Table C-21 lists the USR VSAs. The vendor ID for USR VSAs is 429.
<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>USR-DTE-Data-Idle-Timeout</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>2</td>
<td>USR-Default-DTE-Data-Rate</td>
<td>ENUM:</td>
<td>1-54</td>
</tr>
<tr>
<td></td>
<td></td>
<td>110_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>300_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>600_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1200_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2400_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4800_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>7200_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>9600_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>12K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>14.4K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>16.8_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>19.2K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>38.4K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>75_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>450_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNKNOWN_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>57.6K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>21.6K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>24K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>26K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>28K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>115K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>31K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>33K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>25333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>110_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>300_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>600_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1200_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2400_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>26666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>28000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>29333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>30666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>32000_BPS</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-21 USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>USR-Default-DTE-Data-Rate</td>
<td>33333_BPS</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>34666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>36000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>37333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>38666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>40000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>41333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>42666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>44000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>45333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>46666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>48000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>49333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>50666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>52000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>53333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>54666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>56000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>57333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>58666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>60000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>61333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>62666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>64000_BPS</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>USR-Last-Number-Dialed-Out</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>4</td>
<td>USR-Sync-Async-Mode</td>
<td>ENUM:</td>
<td>1-2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Asynchronous</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Synchronous</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>USR-Originate-Answer-Mode</td>
<td>ENUM:</td>
<td>1-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Originate_in_Originate_Mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Originate_in_Answer_Mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Answer_in_Originate_Mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Answer_in_Answer_Mode</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>USR-Failure-to-Connect-Reason</td>
<td>ENUM:</td>
<td>1-67</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>USR-Initial-Tx-Link-Data-Rate</td>
<td>ENUM:</td>
<td>1-54</td>
</tr>
<tr>
<td></td>
<td></td>
<td>110_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>14.4K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>16.8_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>19.2K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>38.4K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>75_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>450_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNKNOWN_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>57.6K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>21.6K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>24K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>300_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>26K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>28K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>115K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>31K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>33K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>25333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>26666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>28000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>29333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>30666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>600_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>32000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>33333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>34666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>36000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>37333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>38666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>40000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>41333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>42666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>44000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1200_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>45333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>46666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>48000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>49333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>50666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>52000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>53333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>54666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>56000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>57333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2400_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>58666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>60000_BPS</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>USR-Initial-Tx-Link-Data-Rate (continued)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>61333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>62666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>64000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4800_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>7200_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>9600_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>12K_BPS</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>USR-Final-Tx-Link-Data-Rate ENUM:</td>
<td></td>
<td>1-54</td>
</tr>
<tr>
<td></td>
<td></td>
<td>110_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>14.4K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>16.8_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>19.2K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>38.4K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>75_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>450_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNKNOWN_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>57.6K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>21.6K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>24K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>300_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>26K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>28K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>115K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>31K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>33K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>25333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>26666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>28000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>29333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>30666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>600_BPS</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-21 USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>USR-Final-Tx-Link-Data-Rate</td>
<td>32000_BPS</td>
<td>1-54</td>
</tr>
<tr>
<td></td>
<td></td>
<td>33333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>34666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>36000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>37333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>38666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>40000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>41333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>42666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>44000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1200_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>45333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>46666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>48000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>49333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>50666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>52000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>53333_BPS</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>USR-Final-Tx-Link-Data-Rate</td>
<td>54666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>56000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>57333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2400_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>58666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>60000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>61333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>62666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>64000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4800_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>7200_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>9600_BPS</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-21 USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>USR-Modulation-Type</td>
<td>ENUM:</td>
<td>1-28</td>
</tr>
<tr>
<td></td>
<td>USR-Equalization-Type</td>
<td>ENUM:</td>
<td>1-2</td>
</tr>
<tr>
<td>10</td>
<td>USR-Characters-Sent</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>112</td>
<td>USR-Characters-Received</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>13</td>
<td>USR-Blocks-Sent</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>14</td>
<td>USR-Blocks-Received</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>15</td>
<td>USR-Blocks-Resent</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>16</td>
<td>USR-Retrains-Requested</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>17</td>
<td>USR-Retrains-Granted</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>18</td>
<td>USR-Line-Reversals</td>
<td>UINT32</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>USR-Number-Of-Characters-Lost</td>
<td>UINT32</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>USR-Back-Channel-Data-Rate ENUM</td>
<td>450BPS 300BPS None</td>
<td>1-3</td>
</tr>
<tr>
<td>22</td>
<td>USR-Number-of-Blers UINT32</td>
<td>0-0</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>USR-Number-of-Link-Timeouts UINT32</td>
<td>0-0</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>USR-Number-of-Fallbacks UINT32</td>
<td>0-0</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>USR-Number-of-Upshifts UINT32</td>
<td>0-0</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>USR-Number-of-Link-NAKs UINT32</td>
<td>0-0</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>USR-Simplified-MNP-Levels ENUM</td>
<td>Unknown NON_ARQ MNP10ec LAPMAC V42ETC2 V42SREJ PIAFS V120 X75 MNP3 MNP4 V42 HST synchronous MNP2 MNP10(Cellular) V42ETC</td>
<td>0-16</td>
</tr>
</tbody>
</table>
## Vendor-Specific Attributes

### Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>USR-Connect-Term-Reason</td>
<td>ENUM:</td>
<td>1-67</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dtrDrop</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>retransmitLimit</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>linkDisconnectMsgReceived</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>noLoopCurrent</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>invalidSpeed</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>unableToRetrain</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>managementCommand</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>noDialTone</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>keyAbort</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>lineBusy</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>noAnswer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>escapeSequence</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>voice</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>noAnswerTone</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>noCarrier</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>undetermined</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>v42SabmeTimeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>v42BreakTimeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>v42DisconnectCmd</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>v42IdExchangeFail</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>v42BadSetup</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>v42InvalidCodeWord</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>athCommand</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>v42StringToLong</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>v42InvalidCommand</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>none</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>v32Cleardown</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dialSecurity</td>
<td></td>
</tr>
</tbody>
</table>
### Vendor-Specific Attributes

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>USR-Connect-Term-Reason</td>
<td>remoteAccessDenied</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>loopLoss</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ds0Teardown</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>promptNotEnabled</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>noPromptingInSync</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>carrierLoss</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>nonArqMode</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>modeIncompatible</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>noPromptInNonARQ</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dialBackLink</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>linkAbort</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>autopassFailed</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbGenericError</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbLinkErrTxPreAck</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbLinkErrTxTardyACK</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbTransmitBusTimeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>inactivityTimeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbReceiveBusTimeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbLinkErrTxTAL</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbLinkErrRxTAL</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbTransmitMasterTimeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbClockMissing</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbReceivedLsWhileLinkUp</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbOutOfSequenceFrame</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbBadFrame</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbAckWaitTimeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbReceivedAckSeqErr</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>mnpIncompatible</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbReceiveOvrflwRNRFail</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pbReceiveMsgBufOvrflw</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>rcvdGatewayDiscCmd</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>tokenPassingTimeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dspInterruptTimeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>mnpProtocolViolation</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>USR-Connect-Term-Reason</td>
<td>class2FaxHangupCmd</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>hstSpeedSwitchTimeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>undefined</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>remotePassword</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>linkPassword</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>USR-DTR-False-Timeout</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>30</td>
<td>USR-Fallback-Limit</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>31</td>
<td>USR-Block-Error-Count-Limit</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>32</td>
<td>USR-Simplified-V42bis-Usage</td>
<td>ENUM:</td>
<td>1-3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>None</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ccittV42bis</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>mnpLevel5</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>33</td>
<td>USR-DTR-True-Timeou</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>34</td>
<td>USR-Last-Number-Dialed-In-DNIS</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>35</td>
<td>USR-Last-Callers-Number-ANI</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>36</td>
<td>USR-Mbi-Ct-PRI-Card-Slot</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>37</td>
<td>USR-Mbi-Ct-TDM-Time-Slot</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>38</td>
<td>USR-Mbi-Ct-PRI-Card-Span-Line</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>39</td>
<td>USR-Mbi-Ct-BChannel-Used</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>40</td>
<td>USR-IP-Input-Filter</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>41</td>
<td>USR-IPX-Input-Filter</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>42</td>
<td>USR-IP-Output-Filter</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>43</td>
<td>USR-IPX-Output-Filter</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>44</td>
<td>USR-SAP-Output-Filter</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>45</td>
<td>USR-VPN-ID</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>46</td>
<td>USR-VPN-Name</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>47</td>
<td>USR-VPN-Neighbor</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>48</td>
<td>USR-Framed-Routing-V2</td>
<td>ENUM: RIP-V2-Off RIP-V2-On</td>
<td>1-2</td>
</tr>
<tr>
<td>49</td>
<td>USR-VPN-Gateway</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>50</td>
<td>USR-Tunnel-Authenticator</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>51</td>
<td>USR-Packet-Index</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>52</td>
<td>USR-Cutoff</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>53</td>
<td>USR-Access-Accept-Packet</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>54</td>
<td>USR-Primary-DNS-Server</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>55</td>
<td>USR-Secondary-DNS-Server</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>56</td>
<td>USR-Primary-NBNS-Server</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>57</td>
<td>USR-Secondary-NBNS-Server</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>58</td>
<td>USR-Syslog-Tap</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>59</td>
<td>USR-Chassis-Call-Slot</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>60</td>
<td>USR-Chassis-Call-Span</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>61</td>
<td>USR-Chassis-Call-Channel</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>62</td>
<td>USR-Keypress-Timeout</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>63</td>
<td>USR-Unauthenticated-Time</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>64</td>
<td>USR-Bearer-Capabilities</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
</tbody>
</table>
### Table C-21 USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>USR-Speed-Of-Connection</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>66</td>
<td>USR-Max-Channels</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>67</td>
<td>USR-Channel-Expansion</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>68</td>
<td>USR-Channel-Decrement</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>69</td>
<td>USR-Expansion-Algorithm</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>70</td>
<td>USR-Compression-Algorithm</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>71</td>
<td>USR-Receive-Acc-Map</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>72</td>
<td>USR-Transmit-Acc-Map</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>73</td>
<td>USR-Compression-Reset-Mode</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>74</td>
<td>USR-Min-Compression-Size</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>75</td>
<td>USR-IP</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>76</td>
<td>USR-IPX</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>77</td>
<td>USR-Filter-Zones</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>78</td>
<td>USR-Appletalk</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>79</td>
<td>USR-Bridging</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>80</td>
<td>USR-Spoofing</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>81</td>
<td>USR-Host-Type</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>82</td>
<td>USR-Send-Name</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>83</td>
<td>USR-Send-Password</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>84</td>
<td>USR-Start-Time</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>85</td>
<td>USR-End-Time</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>86</td>
<td>USR-Send-Script1</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>87</td>
<td>USR-Reply-Script1</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>88</td>
<td>USR-Send-Script2</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>89</td>
<td>USR-Reply-Script2</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>90</td>
<td>USR-Send-Script3</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>91</td>
<td>USR-Reply-Script3</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>92</td>
<td>USR-Send-Script4</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>93</td>
<td>USR-Reply-Script4</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>94</td>
<td>USR-Send-Script5</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>95</td>
<td>USR-Reply-Script5</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>96</td>
<td>USR-Send-Script6</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>97</td>
<td>USR-Reply-Script6</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>98</td>
<td>USR-Terminal-Type</td>
<td>String</td>
<td>1-253</td>
</tr>
</tbody>
</table>
### Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>99</td>
<td>USR-Appletalk-Network-Range</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>100</td>
<td>USR-Local-IP-Address</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>101</td>
<td>USR-Routing-Protocol</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>102</td>
<td>USR-Modem-Group</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>103</td>
<td>USR-IPX-Routing</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>104</td>
<td>USR-IPX-Wan</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>105</td>
<td>USR-IP-RIP-Policies</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>106</td>
<td>USR-IP-RIP-Simple-Auth-Password</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>107</td>
<td>USR-IDS0-Call-Type</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>108</td>
<td>USR-Call-Terminate-in-GMT</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>109</td>
<td>USR-Call-Connect-in-GMT</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>110</td>
<td>USR-Call-Arrival-in-GMT</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>111</td>
<td>USR-Channel-Connected-To</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>112</td>
<td>USR-Slot-Connected-To</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>113</td>
<td>USR-Device-Connected-To</td>
<td>ENUM: None isdnGateway quadModem</td>
<td>1-3</td>
</tr>
<tr>
<td>114</td>
<td>USR-NFAS-ID</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>115</td>
<td>USR-Q931-Call-Reference-Value</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>SubAttr</td>
<td>VSA Name</td>
<td>Type</td>
<td>Min-Max Value</td>
</tr>
<tr>
<td>---------</td>
<td>-------------------------------</td>
<td>-----------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>116</td>
<td>USR-Call-Event-Code</td>
<td>ENUM:</td>
<td>1-28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>notSupported</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>noFreeIGW</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>igwRejectCall</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>igwSetupTimeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>noFreeTdmts</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>bcReject</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ieReject</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>chidReject</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>progReject</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>callingPartyReject</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>calledPartyReject</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>setup</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>blocked</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>analogBlocked</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>digitalBlocked</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>outOfService</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>busy</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>congestion</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>protocolError</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>noFreeBchannel</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>inOutCallCollision</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>usrSetup</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>telcoDisconnect</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>usrDisconnect</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>noFreeModem</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>modemsNotAllowed</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>modemsRejectCall</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>modemSetupTimeout</td>
<td></td>
</tr>
<tr>
<td>117</td>
<td>USR-DS0</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>118</td>
<td>USR-DS0s</td>
<td>String</td>
<td>1-253</td>
</tr>
<tr>
<td>119</td>
<td>USR-Gateway-IP-Address</td>
<td>IP Address</td>
<td>0-0</td>
</tr>
<tr>
<td>120</td>
<td>USR-Physical-State</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>121</td>
<td>USR-Chassis-Temp-Threshold</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
</tbody>
</table>
### Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>122</td>
<td>USR-Card-Type</td>
<td>ENUM: SlotEmpty</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadV32DigitalModemNAC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DualT1NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DualAlogMdmNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadDgtlMdmNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadAlogDgtlMdmNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>TokenRingNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SingleT1NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>EthernetNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ShortHaulDualT1NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DualAlogMgdIntlMdmNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>X25NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>USR-Card-Type (continued)</td>
<td>ENUM: QuadAlogNonMgdMdmNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadAlogMgdIntlMdmNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadAlogNonMgdIntlMdmNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadLsdLiMgdMdmNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadLsdLiNonMgdMdmNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadLsdLiMgdIntlMdmNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadLsdLiNonMgdIntlMdmNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>EthernetWithV35NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>HSEthernetWithoutV35NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DualHighSpeedV35NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadV35RS122LowSpeedNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DualE1NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ShortHaulDualE1NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>BellcoreLongHaulDualT1NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>BellcoreShrtHaulDualT1NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SCSIEdgeServerNIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadV32AnalogModemNAC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadV32DigAnlModemNAC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadV34DigModemNAC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadV34AnlModemNAC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>QuadV34DigAnlModemNAC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SingleT1NIC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>EthernetGatewayNAC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>AccessServer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>486TrGatewayNAC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SlotUnknown</td>
<td></td>
</tr>
<tr>
<td>SubAttr</td>
<td>VSA Name</td>
<td>Type</td>
<td>Min-Max Value</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------</td>
<td>----------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>122</td>
<td>USR-Card-Type (continued)</td>
<td>ENUM: 486EthernetGatewayNAC, DualRS122NAC, 486X25GatewayNAC, ApplicationServerNAC, ISDNGatewayNAC, ISDNpriT1NAC, CkldNetMgtCard, ModemPoolManagementNAC, NetwMgtCard, ModemPoolNetserverNAC (continued)</td>
<td>1-1027</td>
</tr>
<tr>
<td>122</td>
<td>USR-Card-Type (continued)</td>
<td>ModemPoolV34ModemNAC, ModemPoolISDNNAC, NTServerNAC, QuadV34DigitalG2NAC, QuadV34AnalogG2NAC, QuadV34DigAnlgG2NAC, NETServerFrameRelayNAC, NETServerTokenRingNAC, X2524ChannelNAC, DualT1NAC, WirelessGatewayNac, EnhancedAccessServer, EnhancedISDNGatewayNAC, DualModemNAC, QuadModemNAC, TrGatewayNAC, X25GatewayNAC, DualV34ModemNAC</td>
<td></td>
</tr>
<tr>
<td>112</td>
<td>USR-Security-Login-Limit</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>124</td>
<td>USR-Security-Resp-Limit</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>125</td>
<td>USR-Packet-Bus-Session</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>126</td>
<td>USR-DTE-Ring-No-Answer-Limit</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
</tbody>
</table>
### Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>127</td>
<td>USR-Final-Rx-Link-Data-Rate</td>
<td>ENUM:</td>
<td>1-54</td>
</tr>
<tr>
<td></td>
<td></td>
<td>110_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>14.4K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>16.8_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>19.2K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>38.4K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>75_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>450_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNKNOWN_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>57.6K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>21.6K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>24K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>300_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>6K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>28K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>115K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>31K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>33K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>25333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>26666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>28000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>62666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>9333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>30666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>600_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(continued)</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>127</td>
<td>USR-Final-Rx-Link-Data-Rate (continued)</td>
<td>32000_BPS, 33333_BPS, 34666_BPS, 36000_BPS, 37333_BPS, 38666_BPS, 40000_BPS, 41333_BPS, 42666_BPS, 44000_BPS, 1200_BPS, 45333_BPS, 46666_BPS, 48000_BPS, 49333_BPS, 50666_BPS, 52000_BPS, 53333_BPS, 54666_BPS, 56000_BPS, 57333_BPS, 2400_BPS, 58666_BPS, 60000_BPS, 61333_BPS, 64000_BPS, 800_BPS, 7200_BPS, 9600_BPS, 12K_BPS</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>USR-Initial-Rx-Link-Data-Rate</td>
<td>ENUM:</td>
<td>1-54</td>
</tr>
<tr>
<td></td>
<td></td>
<td>110_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>14.4K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>16.8_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>19.2K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>38.4K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>75_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>450_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>UNKNOWN_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>57.6K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>21.6K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>24K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>300_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>26K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>28K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>115K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>31K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>33K_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>25333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>26666_BPS</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>USR-Initial-Rx-Link-Data-Rate</td>
<td>28000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>29333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>30666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>600_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>32000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>33333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>34666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>36000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>37333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>38666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>40000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>41333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>42666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>44000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>45333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>46666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>48000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>49333_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>50666_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>52000_BPS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>53333_BPS</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>USR-Initial-Rx-Link-Data-Rate</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>54666_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>56000_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>57333_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2400_XBPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>58666_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>60000_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>61333_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>62666_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>64000_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4800_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>7200_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>9600_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>12K_BPS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>129</td>
<td>USR-Event-Date-Time</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>130</td>
<td>USR-Chassis-Temperature</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>131</td>
<td>USR-Actual-Voltage</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>132</td>
<td>USR-Expected-Voltage</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>133</td>
<td>USR-Power-Supply-Number</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>134</td>
<td>USR-Channel</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>135</td>
<td>USR-Chassis-Slot</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
</tbody>
</table>
### Table C-21 USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>136</td>
<td>USR-Event-Id</td>
<td>ENUM:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>HUB_Temp_Out_of_Range</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Fan_Failed</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Watchdog_Timeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Mgmt_Bus_Failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>In_Connection_Est</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Out_Connection_Est</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>In_Connection_Term</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Out_Connection_Term</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Connection_Failed</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Timeout</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DTE_Transmit_Id</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DTR_True</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DTR_False</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Block_Error_at_Threshold</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Fallbacks_at_Threshold</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>No_Dial_Tone_Detected</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>No_Loop_Current_Detected</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Yellow_Alarm</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Red_Alarm</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Loss_Of_Signal</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rcv_Alm_In_Signal</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Timing_Source_Switch</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Modem_Reset_by_DTE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Modem_Ring_No_Answer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DTE_Ring_No_Answer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pkt_Bus_Session_Active</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pkt_Bus_Session_Congestion</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pkt_Bus_Session_Lost</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pkt_Bus_Session_Inactive</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>User_Interface_Reset</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gateway_Port_Out_of_Service</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gateway_Port_Link_Active</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dial_Out_Login_Failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dial_In_Login_Failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dial_Out_Call_Duration</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dial_In_Call_Duration</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Pkt_Bus_Session_Err_Status</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NMC_AutoResponse_Trap</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Continued)</td>
<td></td>
</tr>
</tbody>
</table>
### Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>136</td>
<td>USR-Event-Id (Continued)</td>
<td>Acct_Server_Contact_Loss</td>
<td>6-84</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Yellow_Alarm_Clear</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Red_Alarm_Clear</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Loss_Of_Signal_Clear</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rev_Alm_In_Signal_Clear</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Incoming_Connection_Establish</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Module_Inserted</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Outgoing_Connection_Establish</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Incoming_Connection_Terminate</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Outgoing_Connection_Terminate</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Connection_Attempt_Failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Continuous_CRC_Alarm</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Continuous_CRC_Alarm_Clear</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Physical_State_Change</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Module_Removed</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gateway_Network_Failed</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gateway_Network_Restored</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Packet_Bus_Clock_Lost</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Packet_Bus_Clock_Restored</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>D_Channel_In_Service</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>D_Channel_Out_of_Service</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DS0s_In_Service</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>DS0s_Out_of_Service</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>T1/T1PRI/E1PRI_Call_Event</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>PSU_Voltage_Alarm</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Psu_Incompatible</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>T1,T1-E1/PRI-Call-Arrive-Even</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>T1,T1-E1/PRI-Call-Connect-Even</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>T1,T1-E1/PRI-Call-Failed-Even</td>
<td></td>
</tr>
<tr>
<td>137</td>
<td>USR-Number-of-Rings-Limit</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>138</td>
<td>USR-Connect-Time-Limit</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>139</td>
<td>USR-Call-End-Date-Time</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>140</td>
<td>USR-Call-Start-Date-Time</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>141</td>
<td>USR-Server-Time</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
</tbody>
</table>
### USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>142</td>
<td>USR-Request-Type</td>
<td>ENUM: Access-Request</td>
<td>1-255</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Access-Challenge</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Status-Server</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Status-Client</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Access-Accept</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Access-Reject</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Accounting-Request</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Accounting-Response</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Access-Password-Change</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Access-Password-Ack</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Access-Password-Reject</td>
<td></td>
</tr>
<tr>
<td>143</td>
<td>USR-Old-Password</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>144</td>
<td>USR-Expiration</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>145</td>
<td>USR-Prompt</td>
<td>UINT32</td>
<td>0-1</td>
</tr>
<tr>
<td>146</td>
<td>USR-Char-Noecho</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>147</td>
<td>USR-User-Group-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>148</td>
<td>USR-Call-Reference-Number</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>149</td>
<td>USR-Dial-In-Sec-Mode</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>150</td>
<td>USR-Req-Db-Mdm-Sel</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>151</td>
<td>USR-Req-Db-Login-Valid</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>152</td>
<td>USR-Dialback-Group-Names</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>153</td>
<td>USR-Dial-In-Call-Rest</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>154</td>
<td>USR-Dial-Out-Call-Rest</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>155</td>
<td>USR-Logins-Before-Blacklist</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>156</td>
<td>USR-Failed-Logins</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>157</td>
<td>USR-Allowed-DB-Modems</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>158</td>
<td>USR-VPN-Encrypter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>159</td>
<td>USR-Acct-VPN-Gateway</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>160</td>
<td>USR-Re-CHAP-Timeout</td>
<td>UINT32</td>
<td>0-0</td>
</tr>
<tr>
<td>161</td>
<td>USR-RMMIE-Manufacture-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>162</td>
<td>USR-RMMIE-Product-Code</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>163</td>
<td>USR-RMMIE-Serial-Number</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>164</td>
<td>USR-RMMIE-Firmware-Version</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>165</td>
<td>USR-RMMIE-Firmware-Build-Date</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
### Table C-21  USR VSAs (continued)

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>166</td>
<td>USR-RMMIE-Status</td>
<td>ENUM: notEnabledInLocalModem, notDetectedInRemoteModem, ok</td>
<td>1-3</td>
</tr>
<tr>
<td>170</td>
<td>USR-RMMIE-Last-Update-Time</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>171</td>
<td>USR-RMMIE-Last-Update-Event</td>
<td>ENUM: None, initialConnection, retrain, speedShift, plannedDisconnection</td>
<td>1-5</td>
</tr>
<tr>
<td>172</td>
<td>USR-RMMIE-Rcv-Tot-PwrLvl</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>173</td>
<td>USR-RMMIE-Rcv-PwrLvl-3300Hz</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>174</td>
<td>USR-RMMIE-Rcv-PwrLvl-3750Hz</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>175</td>
<td>USR-RMMIE-PwrLvl-NearEcho-Canc</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>176</td>
<td>USR-RMMIE-PwrLvl-FarEcho-Canc</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>177</td>
<td>USR-RMMIE-PwrLvl-Noise- lvl</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>178</td>
<td>USR-RMMIE-PwrLvl-Xmit-Lvl</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>179</td>
<td>USR-IPX-SAP</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>180</td>
<td>USR-MIC</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>181</td>
<td>USR-Call-Tracking-ID</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>182</td>
<td>USR-Log-Filter-Packet</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>183</td>
<td>USR-CCP-Algorithm</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>184</td>
<td>USR-ACCM-Type</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>185</td>
<td>USR-Connect-Speed</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>186</td>
<td>USR-Framed-IP-Address-Pool-Name</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>187</td>
<td>USR-MP-EDO</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>188</td>
<td>USR-Local-Framed-IP-Addr</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>189</td>
<td>USR-IP-RIP-Input-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>190</td>
<td>USR-IP-Call-Input-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>191</td>
<td>USR-IPX-Call-Input-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>192</td>
<td>USR-AT-Input-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>SubAttr</td>
<td>VSA Name</td>
<td>Type</td>
<td>Min-Max Value</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------</td>
<td>-----------</td>
<td>---------------</td>
</tr>
<tr>
<td>193</td>
<td>USR-AT-RTMP-Input-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>194</td>
<td>USR-AT-Zip-Input-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>195</td>
<td>USR-AT-Call-Input-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>196</td>
<td>USR-ET-Bridge-Input-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>197</td>
<td>USR-IP-RIP-Output-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>198</td>
<td>USR-IP-Call-Output-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>199</td>
<td>USR-IPX-RIP-Output-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>200</td>
<td>USR-IPX-Call-Output-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>201</td>
<td>USR-AT-Output-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>202</td>
<td>USR-ET-RTMP-Output-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>203</td>
<td>USR-AT-Zip-Output-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>204</td>
<td>USR-AT-Call-Output-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>205</td>
<td>USR-ET-Bridge-Output-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>206</td>
<td>USR-ET-Bridge-Call-Output-Filter</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>207</td>
<td>USR-IP-Default-Route-Option</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
<tr>
<td>208</td>
<td>USR-MP-EDO-HIPER</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>209</td>
<td>USR-MP-MRRU</td>
<td>UINT32</td>
<td>0-253</td>
</tr>
</tbody>
</table>
**WiMax**

Table C-22 lists the WiMax VSAs. The vendor ID for WiMax VSAs is 24757.

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>HA-IP-MIP4</td>
<td>IPAddress</td>
<td>0-253</td>
</tr>
<tr>
<td>2</td>
<td>HA-IP-MIP6</td>
<td>IPAddress</td>
<td>0-253</td>
</tr>
<tr>
<td>3</td>
<td>GMT-Time-Zone-Offet</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>4</td>
<td>NAP-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>5</td>
<td>NSP-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>6</td>
<td>Hotline-Indicator</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>7</td>
<td>BS-ID</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>

**WISPr**

Table C-23 lists the WISPr VSAs. The vendor ID for WISPr VSAs is 14122.

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WISPr-Location-ID</td>
<td>String</td>
<td>0-65535</td>
</tr>
<tr>
<td>2</td>
<td>WISPr-Location-Name</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>3</td>
<td>WISPr-Logoff-URL</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>4</td>
<td>WISPr-Redirection-URL</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>5</td>
<td>WISPr-Bandwidth-Min-Up</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>6</td>
<td>WISPr-Bandwidth-Min-Down</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>7</td>
<td>WISPr-Bandwidth-Max-Up</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>8</td>
<td>WISPr-Bandwidth-Max-Down</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>9</td>
<td>WISPr-Session-Terminate-Time</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>10</td>
<td>WISPr-Session-Terminate-End-Of-Day</td>
<td>UINT32</td>
<td>0-65535</td>
</tr>
<tr>
<td>11</td>
<td>WISPr-Billing-Class-Off-Service</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
Table C-24 lists the XML VSAs, attributes for XML tags. The vendor ID for XML VSAs is 5842.

Table C-24 XML VSAs

<table>
<thead>
<tr>
<th>SubAttr</th>
<th>VSA Name</th>
<th>Type</th>
<th>Min-Max Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>XML-Address-format-IPv4</td>
<td>IPADDR</td>
<td>0-253</td>
</tr>
<tr>
<td>2</td>
<td>XML-Association</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>3</td>
<td>XML-Request</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>4</td>
<td>XML-Response</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>5</td>
<td>XML-UserId-id_type-subscriber_id</td>
<td>String</td>
<td>0-253</td>
</tr>
<tr>
<td>6</td>
<td>XML-UserIdRequest</td>
<td>String</td>
<td>0-253</td>
</tr>
</tbody>
</table>
Support for REST API in Cisco Prime Access Registrar

This appendix provides information about the REpresentational State Transfer (REST) APIs supported in Cisco Prime Access Registrar. The purpose of this appendix is to provide a developer, system or network administrator, or system integrator with basic guidelines for using the outlined REST APIs within the Prime Access Registrar deployment.

REST API Framework

REST is a resource-based architectural style to create web services. A resource is an object, which could be a user, address, and so on. Each resource is identified by a Unique Resource Identifier (URI) and is manipulated by representations that pass back and forth between client and server. Representations can be in the form of XML, JSON, Plain, TEXT, or HTML. However, Prime Access Registrar supports only the JSON format.

Table D-1 lists the common operations supported in Prime Access Registrar for REST APIs.

<table>
<thead>
<tr>
<th>Method</th>
<th>Crud Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD</td>
<td>Create a resource</td>
</tr>
<tr>
<td>GET</td>
<td>Read and retrieve a representation of a resource</td>
</tr>
<tr>
<td>EDIT</td>
<td>Update an existing representation</td>
</tr>
<tr>
<td>DELETE</td>
<td>Delete a resource</td>
</tr>
</tbody>
</table>

REST API Services

You can use any client for creating the APIs and must pass the following information as inputs for the APIs:

- Content-Type—application/json
- username—username to access the service
- password—password to access the service
Table D-1 lists the REST APIs used in Prime Access Registrar.

### Table D-2: REST API Services

<table>
<thead>
<tr>
<th>Object</th>
<th>Type ID</th>
<th>ADD</th>
<th>EDIT</th>
<th>GET</th>
<th>DELETE</th>
</tr>
</thead>
</table>
## REST API Services (continued)

<table>
<thead>
<tr>
<th>Object</th>
<th>Type ID</th>
<th>ADD</th>
<th>EDIT</th>
<th>GET</th>
<th>DELETE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Snmp</td>
<td>13</td>
<td>http://&lt;hostname&gt;:8080/RESTAPI/service/addobject</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>
Table D-2  REST API Services (continued)

<table>
<thead>
<tr>
<th>Object -&gt; Type ID</th>
<th>ADD</th>
<th>EDIT</th>
<th>GET</th>
<th>DELETE</th>
</tr>
</thead>
<tbody>
<tr>
<td>RemoteODBCSession Server -&gt; 14</td>
<td>http://&lt;hostname&gt;:8080/RESTAPI/service/addobject</td>
<td>Pass the object and the object name as data.</td>
<td>——</td>
<td>——</td>
</tr>
</tbody>
</table>

Table D-3 provides a list of other REST APIs used in Prime Access Registrar.

Table D-3  Other APIs used in Prime Access Registrar

<table>
<thead>
<tr>
<th>Operation</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tacacs Statistics</td>
<td>http://&lt;hostname&gt;:8080/RESTAPI/service/Tacacsstatistics</td>
</tr>
<tr>
<td>DiameterPeerStats</td>
<td>http://&lt;hostname&gt;:8080/RESTAPI/service/DiameterPeerStats</td>
</tr>
<tr>
<td>ClientStats</td>
<td>http://&lt;hostname&gt;:8080/RESTAPI/service/ClientStats</td>
</tr>
<tr>
<td>Reload</td>
<td>http://&lt;hostname&gt;:8080/RESTAPI/service/Reload</td>
</tr>
<tr>
<td>AddUser</td>
<td>http://&lt;hostname&gt;:8080/RESTAPI/service/adduser?UserListName=&lt;nameofUserNameList&gt;</td>
</tr>
<tr>
<td>EditUser</td>
<td>http://&lt;hostname&gt;:8080/RESTAPI/service/edituser?name=&lt;nameofuser&gt;&amp;UserListName=&lt;nameofuserlist&gt;</td>
</tr>
<tr>
<td>DeleteUser</td>
<td>http://&lt;hostname&gt;:8080/RESTAPI/service/deleteuser?name=&lt;nameofuser&gt;&amp;UserListName=&lt;nameofuserlist&gt;</td>
</tr>
</tbody>
</table>
CoA and PoD REST APIs

The Change of Authorization (CoA) and Packet of Disconnect (PoD) API calls allow you to send session reauthentication and session disconnect commands for a specified session.

You can use any client for creating the APIs and must pass the following information as inputs for the APIs:

- **URL**—URL to access the PoD/CoA service. Example:
  - For PoD—http://<hostname>:8080/RESTAPI/service/PoD
  - For CoA—http://<hostname>:8080/RESTAPI/service/CoA
- **Content-Type**—application/json
- **username**—username to access the service
- **password**—password to access the service
- **data**—API body with syntax as listed in *Table D-4*

**Example**
The following example shows a sample API written using cURL client.

```
```

**Note**
REST interface can also be accessed using HTTPS through the 8443 port.

**Table D-4 Parameter and Data Syntax for APIs**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Syntax/Example</th>
</tr>
</thead>
</table>
| with-id                    | "{"parameter":"S21","value":"","type":"with-id"}"
| with-user                  | "{"parameter":"bob","value":"","type":"with-user"}"
| with-key                   | "{"parameter":"bob","value":"","type":"with-key"}"
| with-nas                   | "{"parameter":"localhost","value":"","type":"with-nas"}"
| with-ip-address            | "{"parameter":"192.168.0.4","value":"","type":"with-ip-address"}"
| with-ipx-network           | "{"parameter":"0x6","value":"","type":"with-ipx-network"}"
| with-age                   | "{"parameter":"1S","value":"","type":"with-age"}"
| with-usr-vpn                | "{"parameter":"1","value":"","type":"with-usr-vpn"}"
| with-attribute             | "{"parameter":"Framed-IP-Address","value":"192.168.0.1","type":"with-attribute"}"
| with-Home-Agent            | "{"parameter":"","value":"","type":"with-Home-Agent"}"
| with-IP-Subnet             | "{"parameter":"","value":"","type":"with-IP-Subnet"}"
**Symbols**

/bin/arserver  17-4

**Numerics**

3gpp compliance  20-7

**A**

AAAFileServiceSyncInterval  4-61
AcceptAll  3-26, 4-12, 4-14, 4-17, 4-18, 4-19, 4-20, 4-24
Accepted-Profiles  B-1
Access-Challenge  1-13
Access Registrar
   definition  1-1
   dictionaries  11-1
   internal database  26-1
   objects  1-2, 4-3
   server  3-10, 4-2
Access-Reject  B-13
Access-Request  5-2, 5-7, 5-8
Accounting  7-1
   attributes  1-15
   database  1-1
   definition  1-1
   log file  3-23, 4-14
MaxFileAge  7-3
MaxFileAge format  7-4
MaxFileSize  7-3
MaxFileSize format  7-4
RolloverSchedule  7-3
setting up  7-2

Start  7-1
Stop  7-1
Accounting records  16-11
Accounting-Service  B-2
ACKaccounting  4-58
Acquire-Dynamic-DNS  B-2
Acquire-Home-Agent  B-2
addProfile method  A-2
AdvancedDuplicateDetectionMemoryInterval  4-63, 4-75
agent_server_logs  29-3
AllowEAPRejectAttrs  4-67
AllowRejectAttrs  4-67
AltigaOutgoingScript  11-8
ANAAAOutgoing  11-8
arbug  17-33
aregcmd
   Cisco Prime Access Registrar command  2-1
   command performance  2-3
   commands  2-4, 2-22
      add  2-5
cd  2-5
delete  2-6
dia-stats  2-16
exit  2-6
filter  2-6
find  2-6
help  2-7
insert  2-7
login  2-7
logout  2-7
ls  2-8
next  2-8
Index

prev 2-8
pwd 2-9
query-sessions 2-9
quit 2-9
release-sessions 2-9
reload 2-10
save 2-10
set 2-11
start 2-12
stats 2-12
status 2-15
stop 2-15
trace 2-19
unset 2-21
validate 2-21
definition 2-1
derror codes 2-24
save 6-4
session management commands 4-35
syntax 2-1
ARIsCaseInsensitive 4-65
arserver file 17-4
AscendIncomingScript 11-8
AscendOutgoingScript 11-9
Attribute Dictionary 1-15, 4-78, A-1
methods A-1
put method A-3
Attributes 4-45, C-1
alphabetical list C-1
check item 17-34
numeric list C-4
AttributesToBeLogged 7-6
AttributesToBeReturned 17-7
Authentication-Service 1-8
Authorization 23-15
definition 1-1
Authorization-Service 1-8

B

BackingStoreDiscThreshold 3-85, 3-124, 4-56, 4-61, 4-74, 22-13
BackingStore-Env-Vars B-4
BaseProfile 3-13, 4-4, 4-5
BindName 4-50
BindPassword 4-50
blacklisting IMSI 23-13
Broadcast-Accounting-Packet B-4

C

CabletronOutgoing 11-9
Cache-Attributes-In-Session B-4
Callback-Number 1-15
callsPerSecond 5-11
Case insensitive commands
  see also aregcmd
cd command 2-1
CertificateDBPath 4-62
change directory command
  see also aregcmd
Change of Authorization (CoA) 17-39
CHAP
  Access Request packet 5-3
CHAP_PASSWORD
  attribute type 4-79
Check item attributes 17-34
CIDR notation 3-111, 4-7
CiscoIncoming 11-9
CiscoOutgoing 11-10
Cisco Prime Access Registrar
  backups 26-1
Cisco Subscriber Edge Services Manager 15-1
CiscoWithODAPIncomingScript 11-10, 13-1, 13-3
Classless Inter-Domain Routing 3-111, 4-7
ClassName 4-11
clear method A-2
Client/server model  1-12
Client-Behind-the-Proxy  11-2
Clients
   IP Address  3-111, 4-7
   List  4-75
   Vendor properties  3-111, 4-7
CoA requests  17-39
Command authorization support  17-57
Commands
   eap-trace  9-44
   Tunnel  9-44
config_mcd_server_logs  29-3
ConfigurationError reply message  4-77
Configuration Examples
   Query-Notify feature  17-8
Configuration Objects  2-3
Configuring
   Check item attributes  17-34
   LDAP RemoteServer  21-3
   Local service  6-2
   ODBC RemoteServer  22-7
Configuring CoA requests  17-39
Configuring rules  18-2
containsKey method  A-2
CRB-Prepaid billing
   With SSG  16-15

D
Database
   Cisco Prime Access Registrar backups  26-1
   MCD  6-4
DataSourceConnections  4-51
DDNS  4-67
DefaultAccountingService  3-11, 4-3
DefaultAuthenticationService  1-4, 3-10, 4-3
DefaultAuthorizationService  3-10, 4-3
DefaultReturnedSubnetSizeIfNoMatch  4-64
DefaultSessionManager  3-10, 4-3
DefaultSessionService  3-11, 4-3
Destination-IP-Address  B-4
Destination-port  B-4
DetectOutOfOrderAccountingPacket  3-77, 4-64
DetectOutOfOrderAccountingPackets  4-64
diameter
   Session management  8-22
diameter-eap  8-2
Dictionaries
   Types of  11-1
Dictionary
   Attribute  4-66
Disable-Accounting-On-Off-Broadcast  B-7
DNSLookupAndLDAPRebindInterval  4-51, 21-6
DropPacket.  3-26, 4-12, 4-14, 4-17, 4-18, 4-19, 4-20, 4-24
Dynamic-DNS-HostName  B-7
Dynamic-Search-Filter  B-7
Dynamic-Service-Loop-Limit  B-7
Dynamic-User-Password-Attribute  B-7

E
EAP  4-13, 9-1
   Authentication mechanism  9-1
   Fatal error packet handling  4-75
   SilentDiscard  4-76
EAP-Actual-Identity  B-8
EAP-AKA’  9-6
eap-aka’  3-38
eap-aka-prime (eap-aka’)  9-7
EAP authentication  9-1
EAP-Authentication-Mode  B-8
EapBadMessagePolicy  4-75
EAP-FAST
   Keystores  4-67
EAP-GTC  9-19, 9-20
EAP-LEAP  9-21
EAP-MD5  9-22
EAP-MSChapv2  9-24
EAP-Negotiate 9-23, 9-24
EAP response messages 4-76
EAP-SIM 9-26
EAP-SIM authentication 9-28, 9-30
eap-trace command 9-44
EAP-Transport Level Security 9-31
Easysoft Open Source 22-12
Empty string 2-1
EnableNotifications 3-112, 4-9
EnableRolloverIntelligence 7-5
EntryPoint 4-25
ENUM
  attribute type 4-79
Environment Dictionary 1-4, 1-8, 11-1, 11-3
Environment Dictionary script 11-4
Environment variable
  Accounting-Service B-2
  Acquire-Group-Session-Limit B-2
  Acquire-IP-Dynamic B-2
  Acquire-IP-Per-NAS-Port B-2
  Acquire-IPX-Dynamic B-2
  Acquire-Subnet-Dynamic B-3
  Acquire-User-Session-Limit B-3
  Acquire-USR-VPN B-3
  Allow-Null-Password B-3
  Authentication-Service B-3
  Authorization-Service B-3
  Current-Group-Count B-4
  Dynamic-Search-Path B-7
  Group-Session-Limit B-8
  Ignore-Accounting-Signature B-9
  Incoming-Translation-Groups B-9
  Misc-Log-Msg-Info B-10
  Reject-Reason B-12
  Remote-Server B-12
  Remote-Servers-Tried B-12
  Request-Authenticator B-12
  Request-Type B-12
  Require-User-To-Be-In-Authorization-List B-13
  Response-Type B-13
  Session-Key B-14
  Session-Manager B-14
  Session-Service B-14
  Source-IP-Address B-15
  Trace-Level B-16
  Unavailable-Resource B-16
  Unavailable-Resource-Type B-16
  User Authorization-Script B-16
  User-Group B-16
  User-Group-Session-Limit B-17
  User-Name B-17
  User-Profile B-17
  User-Session-Limit B-17
Error codes
  aregcmd 2-24
  EscapeSpecialCharInUserName 4-51
  ExecCLIDRule 18-16
  ExecDNISRulerule 18-16
  ExecNASIPRule 18-17
  ExecRealmRule 18-15
  ExecTimeRule 18-12, 18-20
Extensible Authentication Protocols 8-1, 9-1, 23-1
Extension points 11-2

F
Failover policy 3-44, 4-18, 4-19, 4-20
fastrules 3-54, 19-1
  CLI 19-2
  GUI 3-54
Fatal error packet 4-75
Fileaccounting
  AttributesToBeLogged 7-6
  EnableRolloverIntelligence 7-5
  FileType 7-5
Filename 4-25
FilenamePrefix 4-14
file service 3-18, 3-23, 4-12, 4-14
Index

FilenamePrefix 4-14
MaxFileAge 4-15
MaxFileSize 4-14
FileType 7-5
Filter 4-50
firstKey method A-2
Force update 17-12
Framed-IP-Address 1-15
Framed Protocol 1-15
FramedRouting 4-43

I
Identifier 4-43
imsi
blacklisting 23-13
IncomingScript 1-7, 3-10, 3-61, 3-108, 4-2, 4-7, 4-10, 4-32, 4-57
IncomingScriptFailed reply message 4-77
IncomingScriptRejectedRequest reply message 4-78
Incoming scripts 1-3, 1-13
Information collection
automatic 17-33
InitEntryPoint 3-19, 4-11, 4-25
InitEntryPointArgs 4-25
InitialBackgroundTimerSleepTime 4-62
InitializeArg 4-11
InitialTimeout 4-58
Input queue
high threshold 24-4
Interfaces properties 4-1
InternalError reply message 4-77
internal script 11-14
internal scripts 3-18, 11-7, 11-14
CLI 11-15
GUI 3-19
IPADDR
attribute type 4-79
IPAddress 3-111, 4-7
IP-Dynamic Resource Manager 4-38
IP-Per-NAS-Port Resource Manager 4-38
IPX-Dynamic Resource Manager 4-38
isEmpty method A-2

J
Java service 4-17
JavaVMOptions 4-64

G
Gateway
Description 4-43
IPAddress 4-43
LocationID 4-43
Name 4-43
SharedSecret 4-43
TunnelRefresh 4-43
get method A-2
Grouping property 18-2
Group service 16-5, 16-13, 16-14
Group-Session-Limit Resource Manager 4-38
GUI
launching 3-1
logging in 3-3
log out 3-4

H
HiddenAttributes 4-5
Home-Agent-Ipv6 4-40
HostName 4-50
Hot configuration 12-6
Hot-lining 17-39
K

KeyStores 4-67

L

LDAP 21-1

hostname 21-3
MultipleServersPolicy 21-2
protocol 4-50
RemoteServers 4-47

ldap

BindName 4-50
BindPassword 4-50
Filter 4-50
HostName 4-50
LDAPToEnvironmentMappings 4-52
LDAPToRadiusMappings 4-52
LimitOutstandingRequests 4-50
MaxOutstandingRequests 4-51
MaxReferrals 4-51
PasswordEncryptionStyle 4-51
ReferralAttribute 4-51
ReferralFilter 4-51
SearchPath 4-50
Timeout 4-50, 4-54
UserPasswordAttribute 4-50
UseSSL 4-52

LDAP Accounting 3-118
LDAP Rebind 21-6

failures 21-6
LDAP RemoteServer 21-3
LDAP server 1-16
LDAP service 21-2
LDAPToCheckItemMappings 4-52, 21-7
LDAPToEnvironmentMappings 4-52, 21-7
LDAPToRadiusMappings 4-52, 21-7
LEAP 9-21

Lightweight Directory Access Protocol 21-1

LimitOutstandingRequests 4-50
local 3-24, 4-18, 8-17
UserList type 3-27, 4-3, 4-18
localhost 5-7
Local Service 6-2
local service 3-27, 4-3, 4-18
LogFileCount 4-63
Log files 29-3
file system 7-4
managing 7-3
LogFileSize 4-63
Logging in 2-7
GUI 3-3
Logging out 2-7
login command 2-7
Login page 3-3
log method A-2
LogServerActivity 4-59

M

m3ua service

map restore data 23-15
Malformed Request reply message 4-77
map restore data 23-15
authorization flow 23-15
CLI 23-17
insert subscriber data 23-16
map restore data authorization 23-15
CLI configuration 23-17
flow 23-15
insert subscriber data structure 23-16
MapSourceIPAddress 11-11
Master-URL-Fragment 17-11, 8-9
MaxFileSize 4-14
MaximumNumberOfRadiusPackets 4-59
MaximumODBCResultSize 4-64
Index

MaximumOutstandingRequests 4-69
MaxOutstandingRequests 4-51
MaxReferrals 4-51
MaxTries 4-58
MCD 26-1
mcdcd.d01-d03 26-2
mcdConfig.txt 26-2
MCD database 6-4
mcddb.dbd 26-2
mcddb.k01-k03 26-2
mcdshadow 26-1
Measurements
  prepaid billing 16-7
Message logging (Linux) 28-3
Microsoft WPS 17-9
MinimumSocketBufferSize 4-62
Mobile Node-Home Agent 20-1
MPLS 13-1
multiple 1-1
MultipleServersPolicy 4-18, 4-19, 4-20, 21-2, 22-6
MVA
  radclient 5-6

O

ODAP
  accounting service 13-8
  address ranges 13-2
  AllowNullPassword property 13-6
  CiscoIncomingScript 13-3
  configuration summary 13-5
  configuring 13-5
  configuring clients 13-15
  configuring Session Managers 13-14
  detailed configuration 13-5
  on-demand address pool 13-1
  Resource Managers 13-9
  service 13-7
  Session Managers 13-8
  userlist 13-5
  users 13-6
  vendor type 13-4
ODBC.ini file 22-2
ODBCDataSource 22-8, 22-11
ODBC RemoteServer 22-7
ODBC service 22-5
ODBCToEnvironmentMappings 22-11
ODBCToRadiusMappings 22-10
ORACLE_HOME 22-2
Oracle Driver
  Easysoft Open Source 22-12
Oracle functions 22-10
order dependent commands
  see also aregcmd
OS paging size 3-96, 4-33
OutagePolicy 3-26, 4-12, 4-14, 4-17, 4-18, 4-19, 4-20, 4-24
OutageScript 3-25, 4-12, 4-14, 4-17, 4-18, 4-19, 4-20, 4-25
OutgoingScript 3-10, 3-61, 3-108, 4-3, 4-8, 4-10, 4-33, 4-57
OutgoingScriptFailed 4-78
OutgoingScriptRejectedRequest 4-78
Outgoing scripts 1-3, 1-8, 1-14
Outgoing-Translation-Groups B-10

N

NAS 1-1, 7-1
NAS IP Address 4-78
NAS-IP-Address 1-15
NAS-Port 1-15
NAS-Vendor-Behind-the-Proxy 11-2
Neighbor 4-43
NetMask 3-112, 4-8
nextKey method A-3
NotificationProperties 3-113, 4-9
NumberOfRemoteUDPServerSockets 4-59
Overview 1-1

P

Packet buffering 7-11
Packet fields 1-15
packet-identifier 5-4
Packet of disconnect 17-36
Paging size
  operating system 4-42
Paging size (operating system) 3-96, 4-33
ParseTranslationGroupsByCLID 18-10, 18-21
ParseTranslationGroupsByDNIS 18-10, 18-20, 18-21
ParseTranslationGroupsByReal 18-20
ParseTranslationGroupsByRealm 18-10
Password
  length of 3-16, 4-4
Password change 17-12
PasswordEncryptionStyle 4-51
PCO-Parse-Client-Outgoing 16-15
PEAP Version 0 9-45
PEAP Version 1 9-50
Performance
  aregcmd 2-3
PhantomSessionTimeOut 3-98, 4-35
Policies
  configuring 18-1
  validation 18-3
Policy 18-1
Policy engine
  attribute translation 18-9
  parsing translation groups 18-10
  reducing overhead 18-13
  time of day access restrictions 18-11
  wildcard support 18-2
Port
  LDAP 4-50
Port 8080 3-1
Ports 4-66

Ports properties 4-1
PPO-Parse-Prepaid-Outgoing 16-16
PPP 1-3, 1-15, 3-11, 4-45
Prepaid
  AA service 16-5, 16-12
  group service 16-5, 16-13, 16-14
Prepaid billing
  measurements 16-7
Protected EAP 9-1
Proxy server 1-15
put method A-3

Q

Query-Notify 17-6
Query-Notify AttributeGroup
  configuration example 17-9
Query-Notify client
  configuration example 17-9
Query-Service B-11
query-sessions command 4-35
quintets-triplets conversion 9-30

R

radclient
  callsPerSecond 5-11
  multivalued attributes 5-6
  syntax 5-1
  testing EAP-TTLS 9-41
  timetest 5-10
radclient commands 9-43
RADIUS
  attribute name 5-5
  attributes C-1
  messages 1-14
  packet type identifier 5-4
  program flow 1-13
Index

protocol 1-12
server 2-3, 2-7, 4-6, 5-3, 11-4
server test tool 5-1
RADIUS_WORKER_THREAD_COUNT 17-4
RADIUS EXtension. See REX
RADIUS packet fields 1-15
RadiusServer object 1-2, 4-1
radius to diameter 8-19
Re-Accounting-Service B-11
ReactivateTimerInterval 4-48
Re-Authentication-Service B-11
Re-Authorization-Service B-11
ReferralAttribute 4-51
ReferralFilter 4-51
RejectAll 3-26, 4-12, 4-14, 4-17, 4-18, 4-19, 4-20, 4-24
Reject-Reason B-12
release-sessions command 4-35
RemoteLDAPServeThreadTimerInterval 4-62
RemoteRadiusServerInterface 4-65
RemoteServer
  ODBC-Accounting 4-56
  prepaid-crb 4-57
RemoteServers 21-2, 22-6
Remote servers
  policy 4-18, 4-19, 4-20
RemoteServer types 4-48
REMOVE_ALL A-3, A-10
remove method A-3
Renewal 17-12
RepIPMaster 12-9
Replication
  archive 12-4
  automatic resynchronization 12-5
  configuration settings 12-6
  data flow 12-3
  data integrity 12-4
  hot configuration 12-6
  hot-standby 12-1
impact on request processing 12-6
RepIPAddress 12-8
RepTransactionArchiveLimit 12-2, 12-8
RepTransactionSyncInterval 12-2, 12-7, 12-10, 12-11
security 12-4
slaves 12-9
slave server 12-3
transaction order 12-5
transaction verification 12-4
Reply Messages 4-76
RepMasterIPAddress 12-9
RepMasterPort 12-9
RepPort 12-8
RepSecret 12-8
RepType 12-7
Request Dictionary 1-13, 11-1
  script 11-3
Request-Type Packets
  Access-Accept B-13
  Access-Challenge B-13
  Access-Reject B-13
  Access-Request B-13
  Accounting-Request B-13
  Accounting-Response B-13
  Ascend-IPA-Allocate B-13
  Ascend-IPA-Release B-13
  Status-Client B-13
  Status-Server B-13
  USR-Enhanced-RADIUS B-13
  USR-NAS-Reboot-Request B-13
  USR-NAS-Reboot-Response B-13
  USR-Resource-Free-Request B-13
  USR-Resource-Free-Response B-13
  USR-Resource-Query-Request B-13
  USR-Resource-Query-Response B-13
RequireNASsBehindProxyBeInClientList 4-60, 4-75
Resource allocation
  dynamic 1-4
  resource manager
Index

3gpp 1-5, 3-100, 4-39
dynamic-dns 1-5, 3-100
group-session-limit 1-4, 3-99
home-agent 1-4, 3-100
home-agent-IPv6 1-4, 3-100
ip-dynamic 1-4, 3-99
ip-per-NAS-port 1-4, 3-99
ipx-dynamic 1-4, 3-99
remote-group-session-limit 1-5, 3-100
remote-ip-dynamic 1-4, 3-100
remote-session-cache 1-5, 3-100
remote-user-session-limit 1-4, 3-100
session-cache 1-5, 3-100
subnet-dynamic 1-4, 3-99
user-session-limit 1-4, 3-100
usr-vpn 1-4, 3-100

Resource Managers 1-5, 4-38, 4-39
  Group-Session-Limit 4-40
  Home-Agent 4-40
  IP-Dynamic 4-40
  IP-Per-NAS-Port 4-41
  IPX-Dynamic 4-41
  subnet-dynamic 4-42
  User-Session-Limit 4-43
  USR-VPN 4-43

Response Dictionary 1-14, 11-1
  script 11-4
  Response-Type B-13
  Resynchronization automatic 12-5
  full 12-5

REX 3-18, 4-11

REX attribute dictionary
  getBytes method A-7
  putBytes method A-9

REX environment dictionary
  allocateMemory A-11
  clear A-11
  containsKey A-11
  firstKey A-11
  get A-12
  isEmpty A-12
  log A-12
  nextKey A-12
  put A-12
  remove A-12
  reschedule A-12
  size A-13
  trace A-13

rex service
  EntryPoint 4-25
  Filename 4-25
  InitEntryPoint 4-25
  InitEntryPointArgs 4-25

RFC
  2866 7-1
  RFC 2138 4-78, C-1
  RFC Compliance 4-67
  RolloverSchedule 7-3
eRoundRobin policy 3-44, 4-18, 4-19, 4-20

Routing requests 18-4
  based on CLID 18-6
  based on DNIS 18-5
  based on NASIP 18-7
  based on realm 18-4
  based on User-Name Prefix 18-8

RPC services 2-4

Rules
  script and attribute requirements 18-3
  standard scripts 18-15

S

Scripting point 11-1
  NAS IncomingScript 11-4

Scripts 11-6
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACMEOutgoingScript</td>
<td>11-8</td>
</tr>
<tr>
<td>adding script definition</td>
<td>11-5</td>
</tr>
<tr>
<td>AltigaIncomingScript</td>
<td>11-8</td>
</tr>
<tr>
<td>ANAAAOutgoing</td>
<td>11-8</td>
</tr>
<tr>
<td>AuthorizePPP</td>
<td>11-9</td>
</tr>
<tr>
<td>AuthorizeService</td>
<td>11-9</td>
</tr>
<tr>
<td>AuthorizeSLIP</td>
<td>11-9</td>
</tr>
<tr>
<td>AuthorizeTelnet</td>
<td>11-9</td>
</tr>
<tr>
<td>choosing the type of script</td>
<td>11-3</td>
</tr>
<tr>
<td>determining goal</td>
<td>11-1</td>
</tr>
<tr>
<td>ExecCLIDRule</td>
<td>11-10</td>
</tr>
<tr>
<td>ExecDNISRule</td>
<td>11-10</td>
</tr>
<tr>
<td>ExecFilterRule</td>
<td>11-10</td>
</tr>
<tr>
<td>ExecRealmRule</td>
<td>11-10, 11-11</td>
</tr>
<tr>
<td>extension points</td>
<td>11-2</td>
</tr>
<tr>
<td>ParseAAARealm</td>
<td>11-11</td>
</tr>
<tr>
<td>ParseAAARealm</td>
<td>11-12</td>
</tr>
<tr>
<td>ParseAASRealm</td>
<td>11-12</td>
</tr>
<tr>
<td>ParseProxyHints</td>
<td>11-12</td>
</tr>
<tr>
<td>ParseServiceAndAAARealmHints</td>
<td>11-12</td>
</tr>
<tr>
<td>ParseServiceAndAAASRealmHints</td>
<td>11-12</td>
</tr>
<tr>
<td>ParseServiceAndAARealmHints</td>
<td>11-13</td>
</tr>
<tr>
<td>ParseServiceAndAASRealmHints</td>
<td>11-13</td>
</tr>
<tr>
<td>ParseServiceAndProxyHints</td>
<td>11-13</td>
</tr>
<tr>
<td>ParseServiceHints</td>
<td>11-13</td>
</tr>
<tr>
<td>ParseTranslationGroupsByCLID</td>
<td>11-13</td>
</tr>
<tr>
<td>ParseTranslationGroupsByDNIS</td>
<td>11-13</td>
</tr>
<tr>
<td>ParseTranslationGroupsByRealm</td>
<td>11-14</td>
</tr>
<tr>
<td>tParseAASRealm</td>
<td>11-12</td>
</tr>
<tr>
<td>tParseProxyHints</td>
<td>11-12</td>
</tr>
<tr>
<td>tParseServiceAndAAARealmHints</td>
<td>11-12</td>
</tr>
<tr>
<td>tParseServiceAndAAASRealmHints</td>
<td>11-12</td>
</tr>
<tr>
<td>tParseServiceAndAARealmHints</td>
<td>11-13</td>
</tr>
<tr>
<td>tParseServiceAndProxyHints</td>
<td>11-13</td>
</tr>
<tr>
<td>types of</td>
<td>1-3</td>
</tr>
<tr>
<td>UseCLIDAsSessionKey</td>
<td>11-14</td>
</tr>
<tr>
<td>USROutgoingScript</td>
<td>11-14</td>
</tr>
<tr>
<td>writing</td>
<td>11-2</td>
</tr>
<tr>
<td>SCTP multihoming</td>
<td>23-22</td>
</tr>
<tr>
<td>SearchScope</td>
<td>4-51</td>
</tr>
<tr>
<td>SelectPolicy</td>
<td>18-1</td>
</tr>
<tr>
<td>Send-PEAP-URI-TLV</td>
<td>17-11</td>
</tr>
<tr>
<td>Server</td>
<td></td>
</tr>
<tr>
<td>master</td>
<td>12-1</td>
</tr>
<tr>
<td>primary</td>
<td>12-1</td>
</tr>
<tr>
<td>secondary</td>
<td>12-1</td>
</tr>
<tr>
<td>Server health</td>
<td>29-8</td>
</tr>
<tr>
<td>Services</td>
<td></td>
</tr>
<tr>
<td>file</td>
<td>3-23, 4-14</td>
</tr>
<tr>
<td>ldap</td>
<td>4-20</td>
</tr>
<tr>
<td>local</td>
<td>3-24, 3-27, 4-3, 4-18, B-17</td>
</tr>
<tr>
<td>proxy requests</td>
<td>4-47</td>
</tr>
<tr>
<td>radius</td>
<td>4-20</td>
</tr>
<tr>
<td>tacacs-udp</td>
<td>4-20</td>
</tr>
<tr>
<td>used for</td>
<td>1-3</td>
</tr>
<tr>
<td>services</td>
<td>3-22, 4-12</td>
</tr>
<tr>
<td>Services objects</td>
<td>3-22, 4-12</td>
</tr>
<tr>
<td>SESM</td>
<td>15-1</td>
</tr>
<tr>
<td>SessionUnavailable reply message</td>
<td>4-77</td>
</tr>
<tr>
<td>Remote Authentication</td>
<td></td>
</tr>
<tr>
<td>definition</td>
<td>1-1</td>
</tr>
<tr>
<td>types of</td>
<td>1-4</td>
</tr>
<tr>
<td>Session Managers</td>
<td>4-33</td>
</tr>
<tr>
<td>Session record size</td>
<td>3-96, 4-33, 4-42</td>
</tr>
<tr>
<td>Setting attributes</td>
<td></td>
</tr>
<tr>
<td>spaces in value</td>
<td>2-12, 6-9</td>
</tr>
<tr>
<td>Shadow backups</td>
<td>26-1</td>
</tr>
<tr>
<td>Shared key</td>
<td></td>
</tr>
<tr>
<td>MN-HA</td>
<td>20-1</td>
</tr>
<tr>
<td>Shared libraries</td>
<td>A-1</td>
</tr>
<tr>
<td>SharedSecret</td>
<td>4-43, 4-57</td>
</tr>
<tr>
<td>Shared secret</td>
<td>3-111, 4-7</td>
</tr>
<tr>
<td>definition</td>
<td>1-13</td>
</tr>
<tr>
<td>Sign up</td>
<td>17-11</td>
</tr>
<tr>
<td>Sign-up URL</td>
<td>17-10</td>
</tr>
</tbody>
</table>

Cisco Prime Access Registrar 7.0.1 User Guide
sigtran-m3ua
   SCTP multihoming  23-22
size method  A-3
SLIP  1-15
SNMP  4-66, 24-1
   Trap recipients  24-8
   traps  24-3
SNMP Configuration
   community string  24-8
SQLDefinition  22-9
SQL queries  22-10
SQLStatement  22-9
SQL syntax restrictions  22-10
SSG  15-1
stats command  B-10
sticky commands  2-8
STRING
   attribute type  4-79
SynthesizeReverseZone  4-67
syslog messages  28-1

T
TACACS+
   command authorization  17-57
   command authorization flow  17-59
tacacs-udp  4-47
Tcl attribute dictionary  A-1, A-2
   addProfile method  A-2
   clear method  A-2
   firstKey method  A-2
   get method  A-2
   isEmpty method  A-2
   log method  A-2
   nextKey method  A-3
   remove method  A-3
   size method  A-3
   trace method  A-3
Tcl scripts  11-6
TerminationAction reply message  4-78
Timeout  4-50, 4-54
timetest  5-10
tMapSourceIPAddress  11-11
tParseAARealm  11-12
tParseServiceAndAARealmHints  11-13
tParseServiceAndAARealmHints  11-13
tParseServiceAndAARealmHints  11-13
TraceFileCount  4-63
trace-file-count  2-21
TraceFileSize  4-63
trace method  A-3
translation
diameter to radius  8-19
Trap recipients  24-8
Traps
carAccountingLoggingFailure  24-6
carInputQueueFull  24-4
carInputQueueNotVeryFull  24-5
carOtherAccServerResponding  24-6
carOtherAuthServerNotResponding  24-6
carOtherAuthServerResponding  24-5
carServerStart  24-4
carServerStop  24-4
   configuring  24-7
   supported  24-4
Trusted ID
   configuration overview  15-2
Trusted Identity  15-1
tunnel command  9-44
TunnelRefresh  4-43

U
UDPPacketSize  4-59
UINT32
   attribute type  4-79
UnableToAcquireResource reply message  4-77
UNDEFINED
attribute type 4-79
UNIX directories 1-2
UnknownUser reply message 4-77
use_challenge parameter 5-3
UseAdvancedDuplicateDetection 4-63, 4-75
UseBinaryPasswordComparison 4-52
UserDefined 4-4
User extensions. See Scripts.
UserGroups
    check item attributes 17-35
UserList 1-3
    check item attributes 17-34
UserNotEnabled reply message 4-77
User objects 1-3
UserPasswordAttribute 4-50
UserPasswordInvalid 4-77
User-Profile B-17
User profiles 1-3
User properties 4-4
UserService 9-22
User-Session-Limit B-17
User-Session-Limit Resource Manager 4-38
UseSSL 4-52
Using SESM with CAR 15-1
USRIncomingScript 11-14
USRIncomingScript-ignoreAccountingSignature 11-14
USR-VPN
    FramedRouting 4-43
    Identifier 4-43
    Neighbor 4-43
USR-VPN Resource Manager 4-38

VENDOR_SPECIFIC
    attribute type 4-79
VendorID 4-79
Vendor specific attributes
    XML C-86
Vendor-specific attributes C-13
    3GPP2 C-13, C-15
    ACC C-22
    Altiga C-27
    Ascend C-30
    Bay Networks C-45
    Cabletron C-46
    Cisco C-48
    Compatible C-51
    Nomadix C-53
    RedCreek C-53, C-56
    Telebit C-59
    WiMax C-85
    WISPr C-85

VHG/PE router 13-1
VRFs 13-2
VSAs C-13

W

WAP 17-6
Windows 95 Registry 1-2
Windows Provisioning Service (WPS) 17-9
Wireless Application Protocol 17-6

X

XML Query Identity 14-2