



Configuring Cisco Prime Access Registrar

This chapter describes how to configure Cisco Prime Access Registrar 8.0. Prime Access Registrar is very flexible. You can choose to configure it in many different ways. In addition, you can write scripts that can be invoked at different points during the processing of incoming requests and/or outgoing responses.

Before you can take advantage of this flexibility, it helps to configure a simple site. This chapter describes that process. It specifically describes a site that has the following characteristics:

- Uses a single user list for all of its users
- Writes all of its accounting information to a file
- Does not use session management to allocate or track dynamic resources

This chapter contains the following sections:

- [Using aregcmd, page 2-1](#)
- [Configuring a Basic Site, page 2-2](#)
- [Configuring Accounting, page 2-15](#)
- [Configuring SNMP, page 2-15](#)

Using aregcmd

To configure Prime Access Registrar, use the **aregcmd** commands, which are command-line based configuration tools. These commands allow you to set any Prime Access Registrar configuration option, as well as, start and stop the Prime Access Registrar RADIUS server and check its statistics.

This topic contains the following sections:

- [General Command Syntax, page 2-1](#)
- [aregcmd Commands, page 2-2](#)

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 a hierarchical file system. You can also supply full path names 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.

The **aregcmd** commands are case *insensitive*, which means that you can use upper or lowercase letters to designate elements. In addition, when you reference existing elements in the configuration, you only need to 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** if no other element at the current level begins with *ad*.

You can use Prime Access Registrar's command completion feature to see what commands are possible from your current directory location in the Prime Access Registrar server hierarchy by pressing the Tab key. You can also press the Tab key after entering a command to see which objects you might want to manage.

The **aregcmd** commands are 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 two single quotes (') or two double quotes ("). In addition, if you use any arguments that contain spaces, make sure to quote the arguments.

aregcmd Commands

The **aregcmd** commands can be grouped into the following categories:

- Navigation commands—navigates within the Prime Access Registrar hierarchy; commands include **cd**, **ls**, **pwd**, **next**, **prev**, **filter**, and **find**.
- Object commands—adds or deletes objects; commands include **add** and **delete**.
- Property commands—changes the value of properties; commands include **set**, **unset**, and **insert**.
- Server commands—manages the server; commands include **save**, **validate**, **start**, **stop**, **reload**, **status**, **stats**, **dia-stats**, and **trace**.
- Application commands—allows user access to the application; commands include **login**, **logout**, **exit**, **quit**, and **help**.
- Session management commands—queries the server about sessions, release active sessions, or count the number of sessions; commands include **query-sessions**, **release-sessions**, and **count-sessions**.

This chapter uses only a few of the above commands to configure the Prime Access Registrar RADIUS server. For more information about all the **aregcmd** commands, see [Chapter 4, "Setting the CPAR Configurable Option."](#)

Configuring a Basic Site

The simplest RADIUS or Diameter server configuration is a site that uses a single user list for all its users, writes its accounting information to a file, and does not use session management to allocate dynamic resources.

To configure such a site, do the following:

1. Run the **aregcmd** command on your Prime Access Registrar machine.
2. Configure the Prime Access Registrar server settings, such as the server name and the server defaults. For example, with RADIUS object.
3. Add users by copying the sample users.
4. Configure the Network Access Server (NAS) clients and proxies that communicate with Prime Access Registrar.
5. Change profile attributes as needed.

6. Save your changes and reload your Prime Access Registrar RADIUS server.

This topic contains the following sections:

- [Running aregcmd, page 2-3](#)
- [Configuring Prime Access Registrar Server Settings, page 2-4](#)
- [Displaying the UserLists, page 2-7](#)
- [Displaying UserGroups, page 2-9](#)
- [Configuring Clients, page 2-10](#)
- [Configuring Profiles, page 2-11](#)
- [Validating and Using Your Changes, page 2-12](#)
- [Testing Your Configuration, page 2-13](#)
- [Troubleshooting Your Configuration, page 2-14](#)

Running aregcmd

aregcmd is the command-line interface program used to configure the Prime Access Registrar server. The **aregcmd** program is located in **\$INSTALL/bin**.

-
- Step 1** Run the **aregcmd** command:

aregcmd

- Step 2** When asked for “Cluster,” press **Enter**.

- Step 3** Enter your administrator name and password.

When you install Prime Access Registrar software, the installation process creates a default administrator called **admin** with the password **aicuser**.

Prime Access Registrar allows you to perform the following:

- Change the administrator’s password. See [Changing the Administrator’s Password, page 2-3](#).
 - Add additional administrators. See [Creating Additional Administrators, page 2-4](#).
-

Changing the Administrator’s Password

The administrator ID **admin** and password **aicuser** are default settings for all releases of Prime Access Registrar software. For security purposes, you should change the password for **admin** at your earliest convenience.

To change the administrator’s password:

-
- Step 1** Use the **cd** command to change to the **Administrators** level. Prime Access Registrar displays the contents of the **Administrators** object.

cd //localhost/Administrators

- Step 2** Use the **cd** command to change to **admin**:

cd admin

```
[ //localhost/Administrators ]
Entries 1 to 1 from 1 total entries
Current filter: <all>
admin/
```

- Step 3** Use the **set** command to change the administrator's password. You enter the password on the command line in readable form, however, Prime Access Registrar displays it as encrypted.

The following example changes the password to 345. You are asked to reenter it for confirmation.

set Password 345

Optionally, use the **set** command to change the description of the **admin** administrator.

set Description local

- Step 4** Use the **ls** command to display the changed admin.

```
ls
```

Creating Additional Administrators

Use the **add** command to add additional administrators.

- Step 1** Use the **cd** command to change to the **Administrators** level:

cd /Administrators

- Step 2** Use the **add** command and specify the name of the administrator, an optional description, and a password.

The following example adds the administrator `jane`, description `testadmin`, and password `123`:

add jane testadmin 123

- Step 3** Use the **ls** command to display the properties of the new administrator:

```
ls
```

Configuring Prime Access Registrar Server Settings

The top level of the Prime Access Registrar RADIUS server is the RADIUS object itself. It specifies the name of the server and other parameters. In configuring this site, you only need to change a few of these properties.

```
[ //localhost/Radius ]
Name = Radius
Description =
Version = 7.2.0.0
```

```
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/
DeviceAccessRules/
GroupServers/
FastRules/
Advanced/
Replication/
```

**Note**

For session managers, user groups, user lists, and profiles, the attributes defined 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.

This topic contains the following sections:

- [Checking the System-Level Defaults, page 2-5](#)
- [Checking the Server's Health, page 2-6](#)
- [Selecting Ports to Use, page 2-6](#)

Checking the System-Level Defaults

Because this site does not use incoming or outgoing scripts, you do not need to change the scripts' properties (IncomingScript and OutgoingScript).

Since the default authentication and authorization properties specify a single user list, you can leave these unchanged as well (DefaultAuthenticationService and DefaultAuthorizationService). And because you have decided to use a file for accounting information, you can leave this property unchanged (DefaultAccountingService).

Session management, however, is on by default (DefaultSessionManager). If you do not want to use session management, you must disable it. Use the **set** command, enter *DefaultSessionManager*, then specify an empty string by entering a set of double quotes:

```
set DefaultSessionManager ""
```

**Note**

When you do not want Prime Access Registrar to monitor resources for user sessions, you should disable session management because using it affects your server performance.

You have now configured some of the properties for the RADIUS server. The next step is to add users.

Checking the Server's Health

To check the server's health, use the **aregcmd status**. The following issues decrement the server's health:

- Rejection of an Access-Request

**Note**

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 a successful response to any packet increments the server's health.

Selecting Ports to Use

By default, Prime Access Registrar uses well-known Linux ports 1812 and 1813 for TCP/IP communications. Prime Access Registrar can be configured to use other ports, if necessary. You can add them to the list of ports to use.

To configure Prime Access Registrar to use ports other than the default ports, complete the following steps:

Step 1 Change directory to **/Radius/Advanced/Ports**.

```
cd /Radius/Advanced/Ports
```

```
[ //localhost/Radius/Advanced/Ports ]
```

Step 2 Use the **add** command (twice) to add ports in pairs. (The **ls** is entered to show the results of the **add** command.)

```
add 1812
```

```
add 1813
```

```
ls
```

```
[ //localhost/Radius/Advanced/Ports ]
Entries 1 to 2 from 2 total entries
Current filter: <all>

1812/
1813/
```

**Note**

After modifying Access Registrar's default ports setting, to continue using the existing ports, you must add them to the list of ports in **/Radius/Advanced/Ports**.

Step 3 Enter the **save** and **reload** commands to affect, validate, and save your modifications to the Prime Access Registrar server configuration.

save

```
Validating //localhost...
Saving //localhost...
```

reload

```
Reloading Server 'Radius'...
Server 'Radius' is Running, its health is 10 out of 10
```

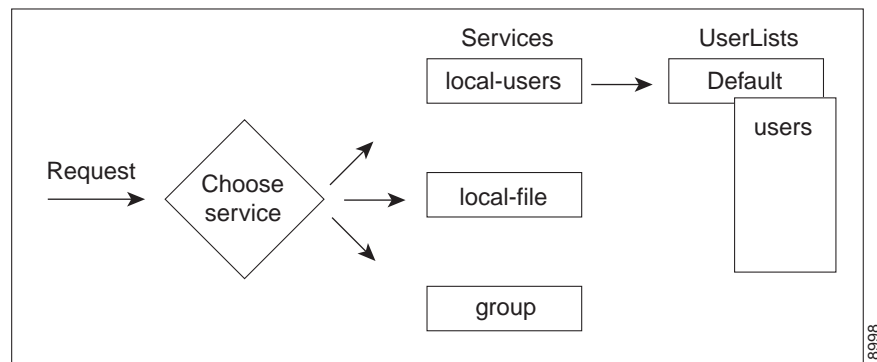
Displaying the UserLists

The first subobject in a RADIUS or Diameter hierarchy that you can configure is the Userlists. The UserLists object contains all of the individual UserLists, which in turn contain the specific users.

When Prime Access Registrar receives an Access-Request, it directs it to an authentication and/or authorization Service. If the Service has its type set to *local*, the Service looks up the user's entry in the specific **UserList**, and authenticates and/or authorizes the user.

Prime Access Registrar, by default, specifies a Service called **local-users** that has the type **local** and uses the **Default** UserList (Figure 2-1).

Figure 2-1 Choosing Appropriate Services



This topic contains the following sections:

- [Displaying the Default UserList, page 2-8](#)
- [Adding Users to UserLists, page 2-8](#)
- [Deleting Users, page 2-9](#)

Displaying the Default UserList

Step 1 Use the **cd** command to change to **UserLists/Default**:

```
cd /Radius/Userlists/Default
```

Step 2 Use the **ls -R** command to display the properties of the three users:

```
ls -R
```

Prime Access Registrar displays the three sample users:

- bob who is configured as a PPP user
 - jane who is configured as a Telnet user
 - joe who is configured as either a PPP or Telnet user depending on how he logs in.
-

Adding Users to UserLists

Use the **aregcmd** command **add** to create a user under a UserList.

To add a user:

Step 1 Use the **add** command to specify the name of a user and an optional description on one command line.

```
add jane
```

```
Added jane
```

Step 2 Change directory to **jane**.

```
cd jane
```

```
[ //localhost/Radius/UserLists/Default/jane ]
Name = jane
Description =
Password = <encrypted>
Enabled = TRUE
Group~ = Telnet-users
BaseProfile~ =
AuthenticationScript~ =
AuthorizationScript~ =
UserDefined1 =
AllowAnonymousPassword = FALSE
Attributes/
```



```
CheckItems/
```

Step 3 Use the **set** command to provide a password for user **jane**.

```
set password jane
```

```
Set Password <encrypted>
```

**Note**

When using the **aregcmd** command, you can use the **add** command and specify all of the properties, or you can use the **add** command to create the object, and then use the **set** command and property name to set the property. For an example of using the **set** command, see the [“Adding a NAS” section on page 2-10](#).

Deleting Users

To delete the sample users, or if you want to remove a user you have added, use the **delete** command. From the appropriate UserList, use the **delete** command, and specify the name of the user you want to delete. For example, to delete user **beth** from the Default UserList, enter:

```
cd /Radius/UserLists/Default
```

```
delete beth
```

Displaying UserGroups

The UserGroups object contains the specific UserGroups. Specific UserGroups allow you to maintain common authentication and authorization attributes in one location, and then have 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.

Prime Access Registrar has three default UserGroups:

- *Default*—uses the script **AuthorizeService** to determine the type of service to provide the user.
- *PPP-users*—uses the BaseProfile **default-PPP-users** to specify the attributes of PPP service to provide the user. The BaseProfile **default-PPP-users** contains the attributes that are added to the response dictionary as part of the authorization. For more information about Profiles, see the [“Configuring Profiles” section on page 2-11](#).
- *Telnet-users*—uses the BaseProfile **default-Telnet-users** to specify the attributes of Telnet service to provide the user. The BaseProfile **default-Telnet-users** contains the attributes that are added to the response dictionary as part of the authorization.

For this basic site, you do not need to change these UserGroups. You can, however, use the **add** or **delete** commands to add or delete groups.

Configuring Clients

The Clients object contains all NAS and proxies that communicate directly with Prime Access Registrar. Each client must have an entry in the Clients list, because each NAS and proxy share a secret with the RADIUS server, which is used to encrypt passwords and to sign responses. See [Adding a NAS, page 2-10](#), for more information on adding a NAS in Prime Access Registrar.

**Note**

If you are just testing Prime Access Registrar with the **radclient** command, the only client you need is **localhost**. The **localhost** client is available in the sample configuration. For more information about using the **radclient** command, see the “[Using radclient](#)” section on [page 2-13](#).

Adding a NAS

You must add your specific NAS from both ends of the connection. That is, you must add Prime Access Registrar for your NAS, and you must add your NAS for Prime Access Registrar.

To add a NAS in Prime Access Registrar:

- Step 1** Use the **cd** command to change to the **Clients** level:

```
cd /Radius/Clients
```

- Step 2** Use the **add** command to add the NAS: `QuickExampleNAS`:

```
add QuickExampleNAS
```

- Step 3** Use the **cd** command to change directory to the **QuickExampleNAS** directory:

```
cd /Radius/Clients/QuickExampleNAS
```

- Step 4** Use the **set** command to specify the description `WestOffice`, the IP address `196.168.1.92`, the shared secret of `xyz`, and the Type as `NAS`.

```
set Description WestOffice
```

```
set IPAddress 209.165.200.225
```

```
set SharedSecret xyz
```

```
set Type NAS
```

```
set Vendor USR
```

```
set IncomingScript ParseServiceHints
```

```
EnableDynamicAuthorization TRUE
```

```
EnableNotifications TRUE
```

The script, **ParseServiceHints**, checks the username for **%PPP** or **%SLIP**. It uses these tags to modify the request so it appears to the RADIUS server that the NAS requested that service.

**Note**

When you are using a different NAS than the one in the example, or when you are adding NAS proprietary attributes, see [Chapter 5, “Configuring and Monitoring the RADIUS Server,”](#) for more information about configuring Client and Vendor objects.

Configure your NAS, using your vendor’s documentation. Make sure both your NAS and the Client specification have the same shared secret.

Configuring Profiles

The Profiles object allows you to set specific RFC-defined attributes that Prime Access Registrar returns in the Access-Accept response. You can use profiles to group 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. The sample users, mentioned earlier in this chapter, reference the following Prime Access Registrar profiles:

- **default-PPP-users**—specifies the appropriate attributes for PPP service
- **default-SLIP-users**—specifies the appropriate attributes for SLIP service
- **default-Telnet-users**—specifies the appropriate attributes for Telnet service.

This topic contains the following sections:

- [Setting RADIUS / Diameter Attributes, page 2-11](#)
- [Adding Multiple Cisco AV Pairs, page 2-12](#)

Setting RADIUS / Diameter Attributes

When you want to set an attribute to a profile, use the following command syntax:

```
set <attribute> <value>
```

This syntax assigns a new value to the named attribute. The following example sets the attribute Service-Type to Framed:

Step 1 Use the **cd** command to change to the appropriate profile and attribute.

```
cd /Radius/Profiles/Default-PPP-users/Attributes
```

Step 2 Use the **set** command to assign a value to the named attribute.

```
set Service-Type Framed
```

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-Routing "192.168.1.0/24 192.168.1.1"
```

Adding Multiple Cisco AV Pairs

When you want to add multiple values to the same attribute in a profile, use the following command syntax:

```
set <attribute> <value1> < value2> < value3>
```

The AV pairs cannot be added one at a time or each subsequent command will overwrite the previous value. For example, consider the following command entry:

```
set Cisco-AVpair "vpdn:12tp-tunnel-password=XYZ" "vpdn:tunnel-type=12tp"  
"vpdn:tunnel-id=telemar" "vpdn:ip-addresses=209.165.200.225"
```

Is

```
Cisco-Avpair = vpdn:12tp-tunnel-password=XYZ  
Cisco-Avpair = vpdn:tunnel-type=12tp  
Cisco-Avpair = vpdn:tunnel-id=telemar  
Cisco-Avpair = vpdn:ip-addresses=209.165.200.225
```



Note

The example above is for explanation only; not all attributes and properties are listed.

Validating and Using Your Changes

After you have finished configuring your Prime Access Registrar server, you must save your changes. Saving your changes causes Prime Access Registrar to validate your changes and, if there were no errors, commit them to the configuration database.

Using the **save** command, however, does not automatically update your server. To update your server you must use the **reload** command. The **reload** command stops your server if it is running, and then restarts the server, which causes Prime Access Registrar to reread the configuration database.

You must **save** and **reload** your configuration changes in order for them to take effect in the Prime Access Registrar server. For more information, see [Saving and Reloading, page 2-12](#).

Saving and Reloading

From anywhere in the radius object hierarchy, enter the **save** and **reload** commands.

Step 1 Use the **save** command to save your changes:

```
save
```

Step 2 Use the **reload** command to reload your server.

```
reload
```

Testing Your Configuration

Now that you have configured some users and a NAS, you are ready to test your configuration. There are two ways you can test your site:

1. You can act as a user and dial in to your NAS, and check that you can successfully log in.
2. You can run the **radclient** command, and specify one of the default users when making a request. For more information, see [Using radclient, page 2-13](#).

Using radclient

You can use the **radclient** command **simple** to create and send a packet. The following example creates an Access-Request packet for user `john` with password `john`, and the packet identifier `p001`. It displays the packet before sending it. It uses the **send** command to send the packet, which displays the response packet object identifier, `p002`. Then, the example shows how to display the contents of the response packet.

Step 1 Run the **radclient** command.

```
./radclient -s
```

Step 2 The **radclient** command prompts you for the administrator's username and password (as defined in the Prime Access Registrar configuration). Use **admin** for the admin name, and **aicuser** for the password.

```
Cisco Prime Access Registrar 7.2.0 RADIUS Test Client
Copyright (C) 1995-2016 by Cisco Systems, Inc. All rights reserved.
Logging in to localhost... done.
```

Step 3 Create a simple Access-Request packet for User-Name `john` and User-Password `john`. At the prompt, enter:

```
simple john john
```

```
p001
```

The **radclient** command displays the ID of the packet `p001`.

Step 4 Enter the packet identifier:

```
p001
```

```
Packet: code = Access-Request, id = 0, length = 0, attributes =
User-Name = john
User-Password = john
NAS-Identifier = localhost
NAS-Port = 0
```

Step 5 Send the request to the default host (**localhost**), enter:

p001 send

p002

Step 6 Enter the response identifier to display the contents of the Access-Accept packet:

p002

```
Packet: code = Access-Accept, id = 1,\
length = 38, attributes =
  Login-IP-Host = 196.168.1.94
  Login-Service = Telnet
  Login-TCP-Port = 541
```

Troubleshooting Your Configuration

If you are unable to receive an Access-Accept packet from the Prime Access Registrar server, you can use the **aregcmd** command **trace** to troubleshoot your problem.

The **trace** command allows you to set the trace level on your server, which governs how much information the server logs about the contents of each packet. You can set the trace levels from zero to four. The system default is zero, which means that no information is logged. For more information, see [Setting the Trace Level, page 2-14](#).

Setting the Trace Level

Step 1 Run the **aregcmd** command.

aregcmd

Step 2 Use the **trace** command to set the trace level to 1-5.

trace 2

Step 3 Try dialing in again.

Step 4 Use the UNIX **tail** command to view the end of the **name_radius_1_trace** log.

```
host% tail -f /opt/CSCOAr/logs/name_radius_1_trace
```

Step 5 Read through the log to see where the request failed.

Configuring Accounting

To configure Prime Access Registrar to perform accounting, you must do the following:

1. Create a service
2. Set the service's 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.

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-20081107-14** would be the 14th rollover on November 07, 2008.

The following shows the properties for a service called Cisco Accounting:

```
[ //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 =
  UseLocalTimeZone = FALSE
```

Configuring SNMP

Before you perform SNMP configuration, you must first stop the operating system-specific SNMP master agent. Then, configure your local **snmpd.conf** file located in the **/cisco-ar/ucd-snmp/share/snmp** directory. Whenever **snmpd.conf** is modified, you must restart the Prime Access Registrar server using the following command for the new configuration to take effect:

```
/etc/init.d/arserver restart
```

This topic contains the following sections:

- [Enabling SNMP in the Cisco Prime Access Registrar Server, page 2-16](#)
- [Stopping the Master Agent, page 2-16](#)
- [Modifying the snmpd.conf File, page 2-16](#)
- [Restarting the Master Agent, page 2-18](#)

Enabling SNMP in the Cisco Prime Access Registrar Server

To enable SNMP on the Prime Access Registrar server, launch **aregcmd** and set the **/Radius/Advanced/SNMP/Enabled** property to TRUE.

```
aregcmd
```

```
cd /Radius/Advanced/SNMP
```

```
[ //localhost/Radius/Advanced/SNMP ]
Enabled = FALSE
TracingEnabled = FALSE
InputQueueHighThreshold = 90
InputQueueLowThreshold = 60
MasterAgentEnabled = TRUE
```

```
set Enabled TRUE
```

Stopping the Master Agent

Stop the Prime Access Registrar SNMP master agent by stopping the Prime Access Registrar server.

```
/opt/CSCOar/bin/arserver stop
```

Modifying the snmpd.conf File

snmpd.conf file is placed in the **/cisco-ar/ucd-snmp/share/snmp** directory. Use **vi** (or another text editor) to edit the **snmpd.conf** file.

The three parts of this file to modify are:

- [Access Control, page 2-16](#)
- [Trap Recipient, page 2-17](#)
- [System Contact Information, page 2-18](#)

Access Control

Access control defines who can query the system. By default, the agent responds to the **public** community for read-only access, if run without any configuration file in place.

The following example from the default **snmpd.conf** file shows how to configure the agent so that you can change the community names, and give yourself write access as well.

To modify the **snmpd.conf** file:

Step 1 Look for the following lines in the **snmpd.conf** file for the location in the file to make modifications:

```
#####
# Access Control
#####
```

Step 2 First map the community name (COMMUNITY) into a security name that is relevant to your site, depending on where the request is coming from:


```
#      sec.name  source      community
# For IPv4
com2sec local    localhost    private
com2sec mynetwork 10.1.9.0/24    public
# For IPv6
com2sec6 local localhost6 public
com2sec6 local 2001:420:27c1:420:214:4fff:fef8:9f3e public
```

The names are tokens that you define arbitrarily.

Step 3 Map the security names into group names:

```
# sec.model  sec.name
group MyRWGroupv1local
group MyRWGroupv2clocal
group MyRWGroupusmlocal
group MyROGroupv1 mynetwork
group MyROGroupv2c mynetwork
group MyROGroupusmmynetwork
```

Step 4 Create a view to enable the groups to have rights:

```
#      incl/excl subtree      mask
view all  included  .1      80
```

Step 5 Finally, grant the two groups access to the one view with different write permissions:

```
#      context  sec.model  sec.level  match  read  write  notif
access MyROGroup ""      any      noauth    exact  all   none   none
access MyRWGroup ""      any      noauth    exact  all   all    none
```

Trap Recipient

The following example provides a sample configuration that sets up trap recipients for SNMP versions v1 and v2c.



Note

Most sites use a single NMS, not two as shown below.

```
# -----
trapcommunity trapcom
trapsink zubat trapcom 162
trap2sink ponyta 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
```

System Contact Information

System contact information is provided in two variables through the **snmpd.conf** file, **syslocation**, and **syscontact**.

Look for the following lines in the **snmpd.conf** file:

```
#####
# System contact information
#
syslocation Your Location, A Building, 8th Floor
syscontact A. Person <someone@somewhere.org>
```

Restarting the Master Agent

Restart the Prime Access Registrar SNMP master agent by restarting the Prime Access Registrar server.

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

Configuring 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 RemoteServers, 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.

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.



Note

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

To configure Dynamic DNS:

-
- 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 the 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
```

```
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.

To test Dynamic DNS using **radclient**:

Step 1 Launch **aregcmd** and log into the Prime Access Registrar server.

```
cd /opt/CSCOar/bin
```

```
aregcmd
```

Step 2 Use the **trace** command to set the trace to level 4.

```
trace 4
```

Step 3 Launch **radclient**.

```
cd /opt/CSCOar/bin
```

```
radclient
```

Step 4 Create an Accounting-Start packet.

```
acct_request Start username
```

Example:

```
set p [ acct_request Start bob ]
```

Step 5 Add a Framed-IP-Address attribute to the Accounting-Start packet.

Step 6 Send the Accounting-Start packet.

\$p send

Step 7 Check the **aregcmd** trace log and the DNS server to verify that the host entry was updated in both the forward and reverse zones.
