



Using On-Demand Address Pools

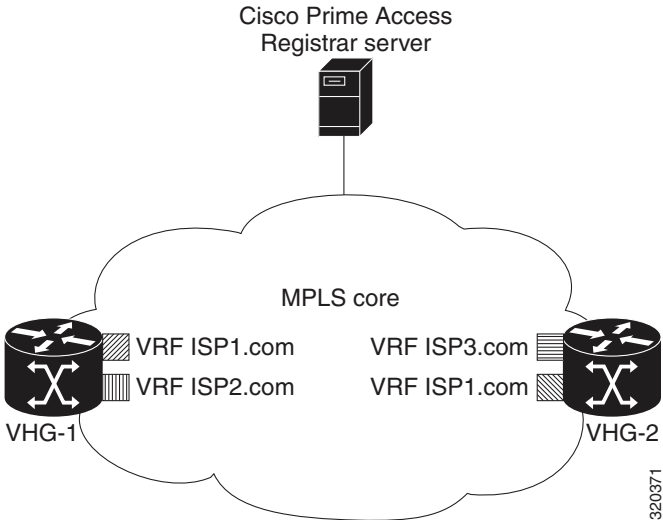
Cisco Prime Access Registrar (Prime Access Registrar) provides support for On-Demand Address Pools (ODAP). Using ODAP, the Prime Access Registrar server manages pools of addresses. Each pool is divided into subnets of various sizes, and the Prime Access Registrar server assigns the subnets to virtual 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, **Cisco**, as its Outgoing Script.

Figure 5-1 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 VHG/PE routers by way of VRFs which contain the subnets and addresses (address space) available.

Figure 5-1 MPLS Core



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 5-1](#), 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
- ...
- 10.10.0.0/16

and:

- 10.255.0.0/24
- 10.255.1.0/24
- ...
- 10.255.10.0/24



Note VRF-ISP2.com requires two ResourceManagers because it has subnets of two different sizes.

- **VRF-ISP3.com**—consists of the address range 172.21.0.0 - 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.

```

proc CiscoWithODAPIncomingScript {request response environ} {

    set RequestType [ $environ get Request-Type ]

    if { [ string compare $RequestType "Access-Request" ] == 0 ||
        [ string compare $RequestType "Accounting-Request" ] == 0 } {

        set NasID [ $request get NAS-Identifier ]

        if { [ string compare $NasID "odap-dhcp" ] == 0 } {
            # Remove the NAS-Identifier - it has done it's job
            $request remove NAS-Identifier

            set UserName [ $environ get User-Name ]
            if { [ string length $UserName ] == 0 } { set UserName [ $request get
User-Name ] }

            # ODAP SUBNET ASSIGNMENT
            $environ put Authentication-Service "odap-users"
            $environ put Authorization-Service "odap-users"
            $environ put Accounting-Service "odap-accounting"
            $environ put Session-Manager "odap-$UserName"

            set AcctSessionId [ $request get Acct-Session-Id ]
            if { [ string length $AcctSessionId ] != 0 } { $environ put Session-Key
$AcctSessionId
            } else {
                $environ log LOG_ERROR "Missing Acct-Session-Id attribute in request-unable
to set Session-Key"
            }
        }
    }
}
CiscoIncomingScript $request $response $environ
}

```

**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.

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, Cisco, as its .

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.

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**.

Configuring Prime Access Registrar to work with ODAP

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**.

Configuring the ODAP Detailed Instructions

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 ]
```

```

Entries 1 to 1 from 1 total entries
Current filter: <all>

Default/

--> add odap-users

Added odap-users

```

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.

```

[ //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.

```

--> cd vrf-ISP2.com

[ //localhost/Radius/UserLists/odap-users/vrf-ISP2.com ]
Name = vrf-ISP2.com
Description =
Password =
Enabled = TRUE

```

```
Group~ =
BaseProfile~ =
AuthenticationScript~ =
AuthorizationScript~ =
UserDefined1 =
AllowNullPassword = FALSE

--> set AllowNullPassword TRUE
```

Setting Up an ODAP-Users Service

Step 4 Add and configure a service for ODAP Users.

```
--> 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
```

Setting Up an ODAP Accounting Service

Step 5 Add and configure an ODAP accounting service.

```
--> 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**.


```

--> 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

- 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/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
```

```
[ //localhost/Radius/ResourceManagers/odap-vrf-ISP3-a.com ]
  Name = odap-vrf-ISP3-a.com
  Description =
  Type =
```

```
--> set type subnet-dynamic
```

```
Set Type subnet-dynamic
```

--> ls

```
[ //localhost/Radius/ResourceManagers/odap-vrf-ISP3-a.com ]
  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

```
[ //localhost/Radius/ResourceManagers/odap-vrf-ISP3-a.com /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

```
[ //localhost/Radius/ResourceManagers/odap-vrf-ISP2-a.com ]
  Name = odap-vrf-ISP2.com
  Description =
  Type =
```

--> set type subnet-dynamic

```
Set Type subnet-dynamic
```

--> ls

```
[ //localhost/Radius/ResourceManagers/odap-vrf-ISP2-a.com ]
  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

```
[ //localhost/Radius/ResourceManagers/odap-vrf-ISP2-a.com /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
```

```

Current filter: <all>

--> add 10.255.0.0-10.255.10.255

Added 10.255.0.0-10.255.10.255

```

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

```

```

[ //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

```

```
[ //localhost/Radius/Clients/vhg-1 ]
  Name = vhg-1
  Description =
  IPAddress = 209.165.200.225
  SharedSecret = secret
  Type = NAS
  Vendor =
  IncomingScript~ =
  OutgoingScript~ =
  UseDNIS = FALSE
  DeviceName = a_name
  DevicePassword = password
```

```
--> set vendor CiscoWithODAP
```

```
Set Vendor CiscoWithODAP
```

Save Your Configuration

Step 10 After completing the configuration, save your changes.

```
--> save
```

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