Cisco IoT Field Network Director Post-Installation Guide - Release 4.3.x

Managing Tunnel Provisioning and High Availability

First Published: October 3, 2018

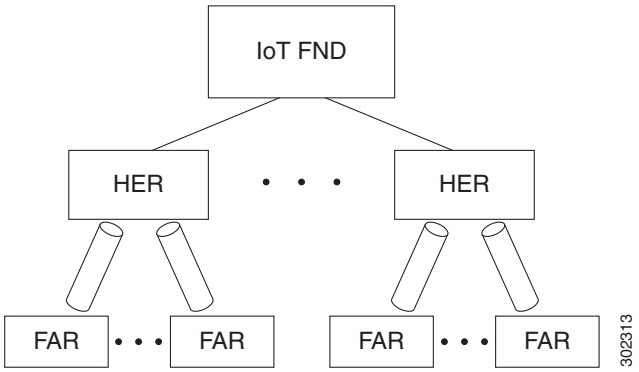
Managing Tunnel Provisioning

This section describes how to configure IoT FND for tunnel provisioning and how to manage and monitor tunnels connecting FARs (CGRs and C800s) and HERs. This section includes the following topics:

Overview

IoT FND sends the commands generated from processing the tunnel provisioning templates to FARs and HERs to provision secure tunnels between them. The default IoT FND templates contain CLI commands to set up and configure GRE and IPsec tunnels. One HER can serve up to 500 FARs, which may include multiple tunnels with the same HER EID and name.

Figure 1 Tunnels Connect FARs and their Corresponding HERs



To provision tunnels between HERs and FARs, IoT FND executes CLI tunnel configuration commands on these devices. By default, IoT FND provides basic tunnel configuration templates containing the CLI tunnel configuration commands. You can also use your own templates. Although the tunnel provisioning process is automatic, you must first complete the configuration steps outlined in Tunnel Provisioning Configuration Process. After that, whenever a FAR comes online, IoT FND automatically provisions it with a tunnel. Before you configure IoT FND for tunnel provisioning, ensure that the IoT FND TPS Proxy is installed and running.

ZTD without IPSec

Beginning with IoT FND Release 3.1.x, you have the option to initiate ZTD with no IPSec configured by ensuring that the Tunnel Provisioning Template is empty of any CLI. This initial approach of bringing up your network without a factory configuration does not preclude subsequent use of IPSec in your network.

Tunnel Provisioning Configuration Process

You must generate the keystore files on the IoT FND and TPS Proxy before configuring tunnel provisioning. Then, you configure IoT FND and the TPS Proxy to talk to one another (refer to *Cisco IoT Field Network Director Installation Guide–Oracle Only Deployment, Release 4.3.x*, “Setting Up the TPS Proxy” and “Configuring IoT FND to Use the TPS Proxy”).

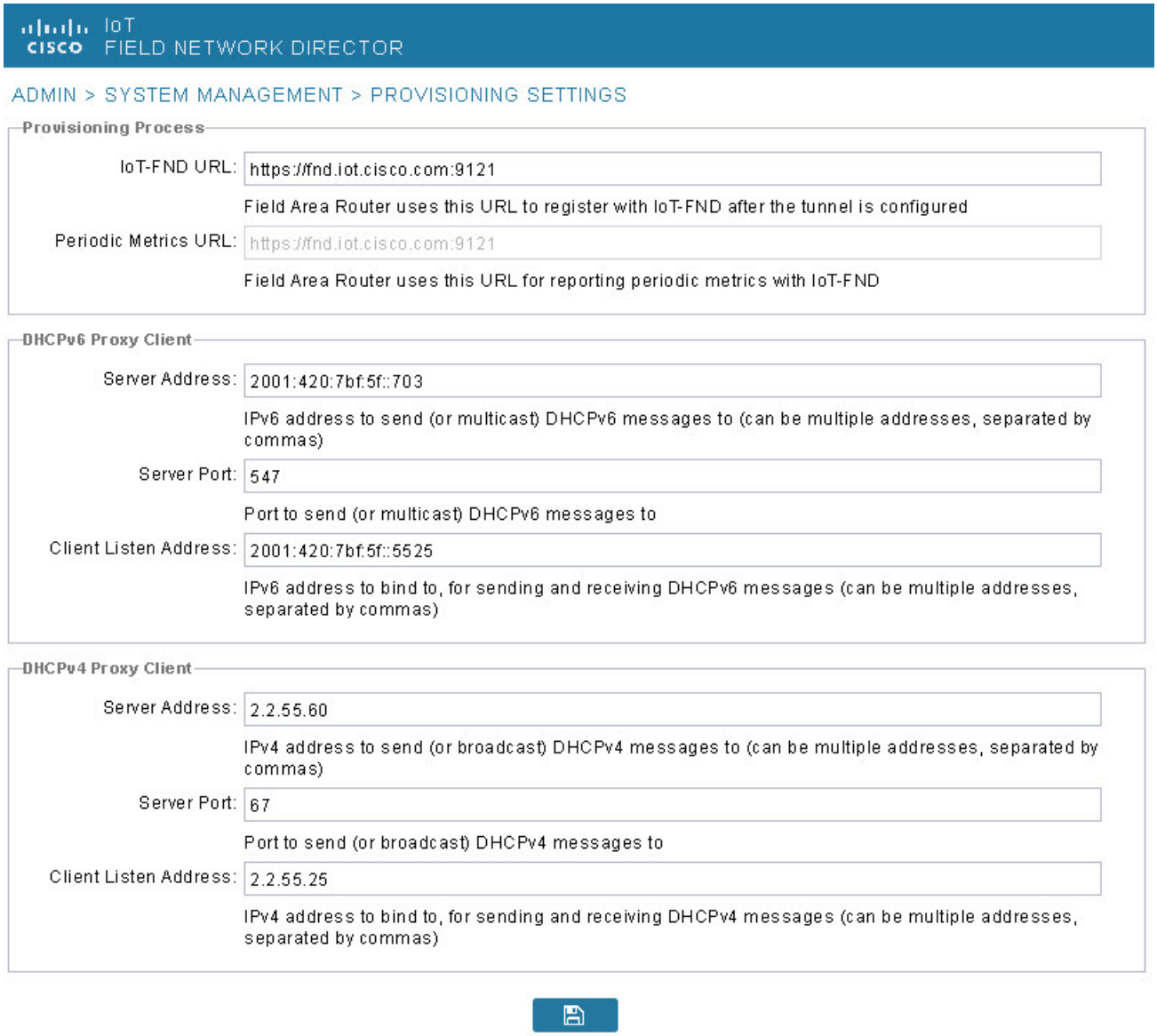
To configure IoT FND for tunnel provisioning:

		Notes
1.	Configure the DHCP servers. Configure DHCP servers to provide unique IP addresses to IoT FND. The default IoT FND tunnel provisioning templates configure a loopback interface and the IP addresses required to create the tunnels. Cisco IOS CGRs use FlexVPN. Ensure that the template only contains addresses for the loopback interface.	Configuring the DHCP Server for Tunnel Provisioning.
2.	Configure the tunnel settings. Configure the NMS URL and the DHCP proxy client settings on the Provisioning Settings page in IoT FND (**Admin > System Management > Provisioning Settings**).	See “Managing System Settings” chapter in Cisco IoT Field Network Director User Guide, Release 4.2.x.
3.	(CG-OS CGRs) Configure IoT FND to accept FAR registration requests on first contact (*call home*) to request tunnel provisioning. Cisco IOS CGRs use the CGNA service.	
4.	Configure HER management. Configure HERs to allow management by IoT FND using NETCONF over SSH.	Configuring HERs Before Adding them to IoT FND. See “Managing Devices” chapter in Cisco IoT Field Network Director User Guide, Release 4.2.x.
5.	Add HERs to IoT FND.	Adding HERs to IoT FND. See “Managing Devices” chapter in Cisco IoT Field Network Director User Guide, Release 4.2.x.
6.	Review the IoT FND tunnel provisioning templates to ensure that they create the correct type of tunnel.	
7.	(Optional) If you plan to use your own templates for tunnel provisioning, create one or more tunnel provisioning groups and modify the default tunnel provisioning templates.	Configuring Tunnel Provisioning Templates
8.	(CG-OS CGRs) Configure FARs to call home. Configure FARs to contact IoT FND over HTTPS through the IoT FND TPS proxy.	This step is typically performed at the factory where the FARs are configured to contact the TPS Proxy.
9.	Add FARs to IoT FND. Import the FARs into IoT FND using the Notice-of-Shipment XML file.	
10.	Map FARs to their corresponding HER.	Tunnel Provisioning Configuration Process

After completing the previous steps, deploy the FARs and power them on. Tunnel provisioning happens automatically.

This is the sequence of events after a FAR is turned on:

1. Upon joining the uplink network after being turned on, the FAR sends a request for certificate enrollment.
2. The FAR then requests tunnel provisioning to IoT FND through the IoT FND TPS Proxy.
3. IoT FND looks up the FAR record in the IoT FND database and determines which tunnel provisioning templates to use. IoT FND also looks up which HERs to which to establish a tunnel.
4. For Cisco IOS CGRs, the default templates configure the CGR to use FlexVPN. The FlexVPN client is configured on the CGR that will contact the HER and ask for a FlexVPN tunnel to be dynamically constructed. This is how the HER dynamically adds a new tunnel endpoint interface for the CGR.
5. Before processing FAR templates, IoT FND processes the HER Tunnel Deletion template and sends the resulting commands to the HERs. This is done for each HER to remove existing tunnel configuration that may be associated with the FAR.
6. IoT FND uses the FreeMarker template engine to process the FAR Tunnel Addition template. The engine converts the templates to text, which IoT FND assumes to be CLI configuration commands (CG-OS or Cisco IOS, per the CGR). IoT FND uses these commands to configure and bring up one end of the tunnel on the FAR.
7. IoT FND uses the FreeMarker template engine to process the HER Tunnel Addition template. The engine converts the templates to text, which IoT FND assumes to be commands for configuring the tunnel on the HERs.
8. This step is OS-specific:
   - For Cisco IOS CGRs, if no errors occurred applying the commands generated by the templates to the FAR and HERs, IoT FND configures a new active CGNA profile “cg-nms-register,” and deactivates the cg-nms-tunnel profile. That cg-nms-register profile uses the IoT FND URL.
   - For CG-OS CGRs, IoT FND re-configures the call home URL to the IoT FND URL specified in the Provisioning Settings page (**ADMIN > System Management > Provisioning Settings**).



The specified URL uses the IoT FND registration port (default 9121) instead of the tunnel provisioning port. The Fully Qualified Domain Name (FQDN) in that URL is different and resolves to an IP address that is only reachable through the tunnels.

Configuring Tunnel Provisioning

This section describes how to configure IoT FND for tunnel provisioning.

- Configuring the DHCP Server for Tunnel Provisioning
- Configuring DHCP for Tunnel Provisioning Using CNR

Configuring the DHCP Server for Tunnel Provisioning

For tunnel provisioning to succeed, configure the DHCP server used by IoT FND to supply addresses to create tunnels between the FARs and HERs. For example, configure the DHCP server to provide IP addresses for tunnel provisioning on a permanent-lease basis.

IoT FND makes the DHCP requests based on the settings defined in the tunnel provisioning templates. During tunnel provisioning, the IoT FND templates can make two kinds of DHCP requests:

- Request an IP address, and then make it available to the template.
- Request a subnet with two IP addresses, and then make both addresses available to the template.

IoT FND can make these requests for IPv4 addresses and IPv6 addresses.

The ability to request DHCP addresses from the template gives you maximum flexibility when defining tunnel configurations because you allocate the exact address needed for each FAR and corresponding interface on the HER. The default tunnel provisioning templates provided address the most common use case: one IPsec tunnel between the FAR and its corresponding HER. Each end of this IPsec tunnel gets a dynamically allocated IPv4 address:

- If your DHCP server supports subnet allocation, use it to obtain two addresses that belong to the same subnet.
- If your DHCP server only supports address allocation, configure it so that the two DHCP address requests return addresses that can be used as ends of an IPsec tunnel.
- If your routing plan calls for allocating unique IPv4 addresses for each FAR and assigning it to a loopback interface above the IPsec tunnel, allocate this address using the IoT FND template.

If you choose to build IPv6 GRE tunnels, allocate the IPv6 addresses for each end of the tunnel using DHCP prefix delegation or individual address requests.

This section describes example DHCP settings for tunnel provisioning. How you configure these settings depends on your installation. This section provides general guidelines for configuring the DHCP server for tunnel provisioning using the Cisco Network Registrar (CNR).

Configuring DHCP for Tunnel Provisioning Using CNR

The CNR CLI script in the following example configures the CNR DHCP server to service requests made by the default tunnel provisioning templates in IoT FND. When using this script, ensure that the subnets are appropriate for your DHCP server environment.

Example CNR DHCP Server Tunnel Provisioning Script

```
# These commented out commands support re-applying the configuration by first
# removing any previously applied configuration, in reverse order. This should
# not be done in a production environment, but may be useful when initially
# developing and testing a configuration.

# scope v4address-perm delete
# dhcp-address-block v4subnet-perm delete
# prefix v6subnet-perm delete
# prefix v6address-perm delete
# policy permanent delete

# Configure the server to automatically map any IPv4 or IPv6 user class
# option values to selection tags. By default CG-NMS includes a value of
# "CG-NMS" for the user class in its requests. The tag is used to insure
# prefixes and scopes configured to satisfy requests from CG-NMS are only
# used for that purpose.

dhcp set map-user-class-id=append-to-tags

# Since CG-NMS uses the leased addresses and subnets in router
# configuration the addresses and subnets must be permanently allocated
# for that purpose. Create a policy that instructs the DHCP server to
# offer a permanent lease.
```

```
policy permanent create
policy permanent set permanent-leases=enabled

# Configure DHCPv6.

# The default CG-NMS tunnel template will request IPv6 addresses for
# use with CGR loopback interfaces.

prefix v6address-perm create 2001:DB8:0:0:1::/80 dhcp-type=dhcp
prefix v6address-perm set description="Pool for leasing addresses for loopback interfaces."
prefix v6address-perm set policy=permanent
prefix v6address-perm set selection-tags=CG-NMS

# The default CG-NMS tunnel template will request IPv6 prefixes for
# use with GRE tunnels. Force use of a /127 prefix.

prefix v6subnet-perm create 2001:DB8:0:0:2::/80 dhcp-type=prefix-delegation
prefix v6subnet-perm set description="Pool for leasing prefixes for GRE tunnels."
prefix v6subnet-perm set policy=permanent
prefix v6subnet-perm set selection-tags=CG-NMS
prefix-policy v6subnet-perm set default-prefix-length=127
prefix-policy v6subnet-perm set shortest-prefix-length=127

# Configure DHCPv4.

# The default CG-NMS tunnel template will request IPv4 subnets for
# use with IPsec tunnels. Note that currently address pools for
# IPv4 subnet allocation can only be configured using the CLI as the
# CNR Web UI does not currently support them.

# If CNR allowed you to set a description on DHCP address blocks it would be:
# "Pool for leasing subnets for IPsec tunnels."

dhcp-address-block v4subnet-perm create 192.0.2.0/24
dhcp-address-block v4subnet-perm set default-subnet-size=31
dhcp-address-block v4subnet-perm set policy=permanent
dhcp-address-block v4subnet-perm set selection-tags=CG-NMS

# The default CG-NMS tunnel template will request IPv4 addresses for
# use with loopback interfaces.

scope v4address-perm create 198.51.100.0 255.255.255.0
scope v4address-perm set description="Pool for leasing addresses for loopback interfaces."
scope v4address-perm set policy=permanent
scope v4address-perm addRange 198.51.100.2 198.51.100.254
scope v4address-perm set selection-tag-list=CG-NMS

# Configure detailed logging of incoming and outgoing packets. This is useful when
# debugging issues involving DHCP, however this level of logging will lower the
# performance of the DHCP server. If this is a production server under heavy load
# it may be necessary to forgo detailed packet logging.

dhcp set
log-settings=missing-options,incoming-packet-detail,outgoing-packet-detail,unknown-criteria,client-deta
il,client-criteria-processing,dropped-waiting-packets,v6-lease-detail

# Save the changes and reload the server to have them take effect.
save
dhcp reload

# List the current configuration.

policy list
prefix list
dhcp-address-block list
```

```
scope list
dhcp show
```

Configuring Tunnel Group Settings

You use groups in IoT FND to bulk configure tunnel provisioning. By default, all FARs are added to the appropriate default group (default-cgr, default-c800). Default groups contain the templates used for tunnel provisioning.

Topics in this section include the following:

Creating Tunnel Groups

If you plan to use one set of templates for all FARs, whether using the default templates, modified default templates or custom templates, do not create additional groups. To define multiple sets of templates, create groups and customize the templates for these groups.

Note: CGRs and C800s can be in the same tunnel provisioning group if your custom templates are applicable to both router types.

To create a tunnel group:

1. Choose **CONFIG > Tunnel Provisioning**.
2. Click **+** icon in left pane to add a group.
3. Enter a name of the new group, and then click **OK**.

   The group appears in the Tunnel Groups pane.

After creating a tunnel group, the next step is to move FARs from other groups to it, as described in Moving FARs to Another Group.

Deleting Tunnel Groups

Only empty groups can be deleted. Before you can delete a tunnel group, you must move the devices it contains to another group.

To delete an empty tunnel group:

1. Choose **CONFIG > Tunnel Provisioning**.
2. In the TUNNEL GROUPS left pane, select the tunnel group to delete.
3. Click (**-**) to delete the group.
4. Click **Yes** to confirm deletion.

Viewing Tunnel Groups

The Tunnel Provisioning page lists information about existing tunnel groups.

Follow these steps to view the tunnel groups defined in IoT FND:

1. Choose **CONFIG > Tunnel Provisioning**.
2. Click **Group Members** tab.
3. In the TUNNEL GROUPS pane (left), select a group.

IoT FND displays the following Tunnel Group information for each router in the group. Not all routers support all fields. (See Table 1).

Table 1 Tunnel Group Fields

Field	Description
Name	Router EID (device identifier).
Status	Status of the router: ■ Unheard—The router has not contacted IoT FND yet. ■ Unsupported—The router is not supported by IoT FND. ■ Up—The router is in operation. ■ Down—The router is turned off.
Last Heard	Last time the router contacted or sent metrics to IoT FND. If the router never contacted IoT FND, **never** appears in this field. Otherwise, IoT FND displays the date and time of the last contact, for example, **4/10 19:06**.
Tunnel Source Interface 1 Tunnel Source Interface 2	Router interface used by the tunnel.
OSPF Area 1 OSPF Area 2	Open shortest path first (OSPF) areas 1 and 2.
OSPFv3 Area 1 OSPFv3 Area 1	OSPFv3 area 1. OSPFv3 area 2.
IPsec Dest Addr 1 IPsec Dest Addr 2	IPv4 destination address of the tunnel.
GRE Tunnel Dest Addr 1 GRE Tunnel Dest Addr 2	IPv6 destination address of the tunnel.
Certificate Issuer Common Name	Name of the CA that issued the certificate.

Renaming a Tunnel Group

You can rename a tunnel group at any time. Cisco recommends using short, meaningful names. Names cannot be more than 250 characters long.

To rename a tunnel group:

1. Choose **CONFIG > Tunnel Provisioning**.
2. In the TUNNEL GROUPS pane, mouse over the tunnel group to rename and click the **Edit** pencil icon ( ).
3. Enter the new Group Name and then click **OK**.

Note: When you enter an invalid character entry (such as, @, #, !, or +) in the entry field, the field is highlighted in red and disables the **OK** button.

Moving FARs to Another Group

You can move FARs to another group in two ways:

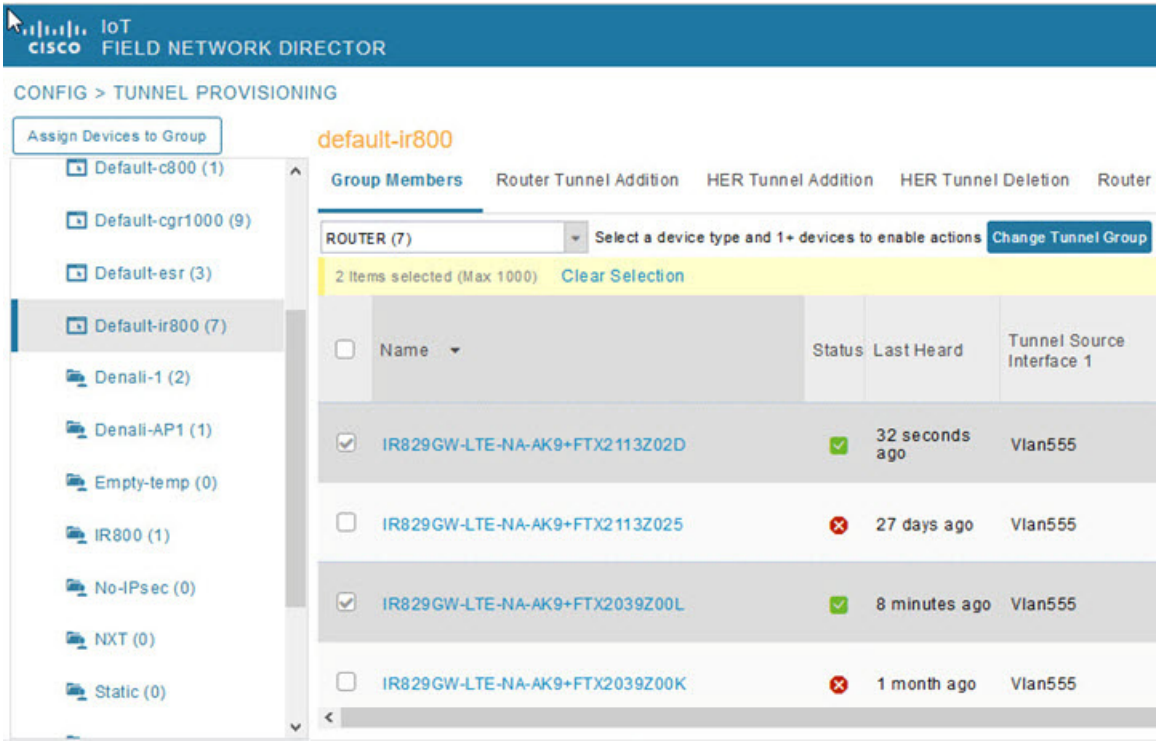
- Moving FARs to Another Group Manually
- Moving FARs to Another Group in Bulk

Moving FARs to Another Group Manually

To move FARs to another group manually:

1. Choose **CONFIG > Tunnel Provisioning**.
2. Click the **Group Members** tab.
3. In the TUNNEL GROUPS pane, select the tunnel group with the routers to move.
4. Choose the device type from the **Select a device type** drop-down menu.
5. Check the check boxes of the FARs to move.

   To select all FARs in a group, click the check box at the top of the column. When you select devices, a yellow bar displays that maintains a count of selected devices and has the Clear Selection and Select All commands. The maximum number of devices you can select is 1000.
6. Click the **Change Tunnel Group** button.



7. From the drop-down menu, choose the tunnel group to which you want to move the FARs.

8. Click **Change Tunnel Group**.
9. Click **OK** to close the dialog box.

Moving FARs to Another Group in Bulk

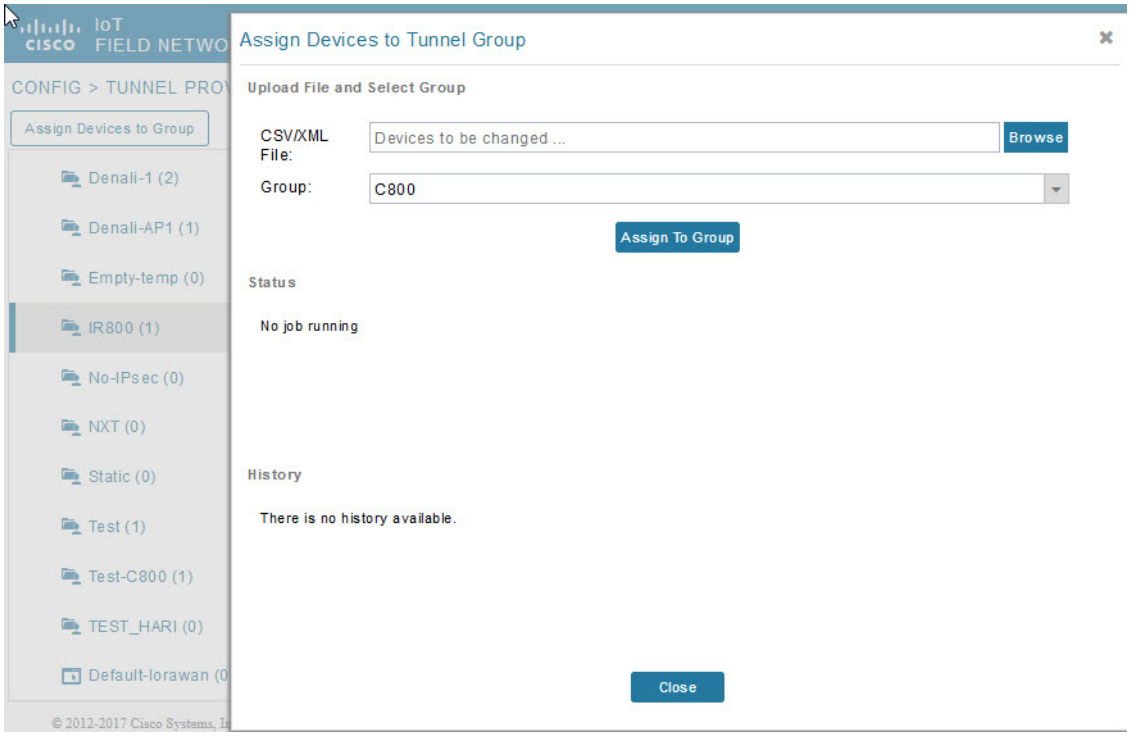
You can move FARs in bulk to another group by importing a CSV or XML file containing the names of the FARs to move. Ensure that the file contains entries in the format shown the following example:

```
eid
CGR1120/k9+JSM1
CGR1120/k9+JSM2
CGR1120/k9+JSM3
CGR1120/k9+JSM4
C819HGW-S-A-K9+FTX174685V0
```

The first line is the header, which tells IoT FND to expect FAR EIDs in the remaining lines (one FAR EID per line).

To move FARs to another group in bulk:

1. Create a CSV or XML file with the EIDs of the devices to move to a different group.
2. Choose **CONFIG > Tunnel Provisioning**.
3. Click **Assign Devices to Tunnel Group** to open an entry panel.



4. Click **Browse** and locate the file that contains the FARs that you want to move.
5. From the **Group** drop-down menu, choose the destination tunnel group.
6. Click **Assign To Group**.
7. Click **Close**.

Configuring Tunnel Provisioning Templates

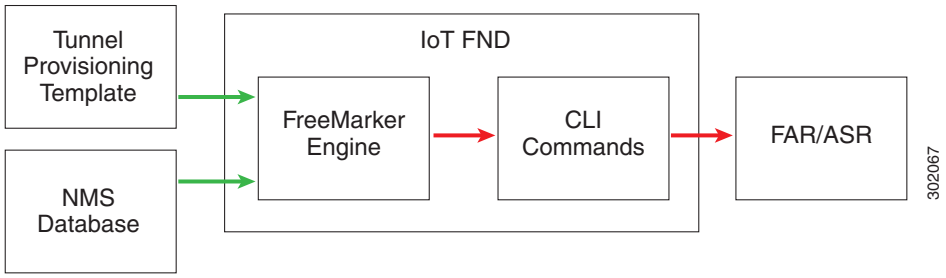
IoT FND has three default tunnel provisioning templates:

- Field Area Router Tunnel Addition—IoT FND uses this template to generate the CLI configuration commands for creating one end of an IPsec tunnel on the FAR.
- Head-End Router Tunnel Addition—IoT FND uses this template to generate the CLI configuration commands for creating the other end of the IPsec tunnel on the HER.
- Head-End Router Tunnel Deletion—IoT FND uses this template to generate the CLI configuration commands for deleting any existing tunnel to the FAR at the other end of the tunnel.

Tunnel Provisioning Template Syntax

The IoT FND tunnel provisioning templates are expressed with the FreeMarker syntax. FreeMarker is an open-source Java-based engine for processing templates and is built into IoT FND. As shown in Figure 2, FreeMarker takes as input the tunnel provisioning template and data supplied by IoT FND, and generates CLI commands that IoT FND runs on the FARs and HERs in the “configure terminal” context.

Figure 2 CLI Command Generation from Templates in IoT FND



In IoT FND, the tunnel provisioning templates consist of router CLI commands and FreeMarker variables and directives. The use of FreeMarker syntax allows IoT FND to define one template to provision multiple routers.

This section describes the basic FreeMarker syntax in the tunnel provisioning templates. For information about FreeMarker visit http://freemarker.sourceforge.net/.

- Template Syntax
- Data Model

Template Syntax

Table 2 describes the syntax in the default tunnel provisioning templates.

Table 2 Tunnel Provisioning Template Syntax

Component	Description
Text	Unmarked text is carried through as CG-OS CLI configuration commands for FARs and Cisco IOS CLI commands for HERs.
Interpolations	${variable} FreeMarker replaces this construct with the value of a string variable that IoT FND supplies. In this example, IoT FND provides the EID of the FAR: `description IPsec tunnel to ${far.eid}`
Default Values	${variable!" Default" } FreeMarker replaces this construct with the value of a string variable. If the variable is not set, FreeMarker replaces this construct with **Default**.

Conditionals	<#if condition> output1 <#else> output2 </#if>
	
	FreeMarker uses this construct to determine the text to use in the output. For example:
	
	`<#if far.ipsecTunnelDestAddr1??>`
	`  <#assign destinationAddress=far.ipsecTunnelDestAddr1>`
	`<#else>`
	`  <#assign destinationAddress=`
	`her.interfaces("GigabitEthernet0/0/0")[0].v4.addresses[0].address>`
	`</#if>`

Iteration over lists	<#list list as variable> ${variable} </#list> FreeMarker uses this construct to iterate over a list.
Comments	<#-- this is a comment --> FreeMarker allows comments, but does not retain them in the output.

Table 2 Tunnel Provisioning Template Syntax (continued)

Component	Description

Assign statements	<#assign name=value>
	
	This construct declares a local variable within the template and assigns a value to it. After that, use this construct to reference the variable:
	
	${name}
	
	For example:
	
	`<#assign interfaceNumber=0>`
	`...`
	`interface Tunnel${interfaceNumber}`

Macros	These constructs are similar to function calls.
	
	<#macro name(param1,param2, ... ,paramN)>
	
	... ${param1} ...
	
	</#macro>
	
	Here is an example of a macro definition:
	
	`<#macro configureTunnel(interfaceNamePrefix,ospfCost)>`
	`  <#assign wanInterface=far.interfaces(interfaceNamePrefix)>`
	`  <#if (wanInterface[0].v4.addresses[0].address)??>`
	`  <#assign interfaceName=wanInterface[0].name>`
	`  interface Tunnel${her.unusedInterfaceNumber()}`
	`  description IPsec tunnel to ${far.eid}`
	`  ...`
	`  ip ospf cost ${ospfCost}`
	`  ...`
	`</#macro>`

Macro calls	To call macros in a tunnel provisioning template: <@name param1, param2 ... paramN> FreeMarker replaces the macro call with the output of the macro after resolving all variables. For example: `<@configureTunnel far.tunnelSrcInterface1!"Wimax", 100/>`

Data Model

This section describes the data model in the tunnel provisioning templates. The **far** and **her** prefixes provide access to the properties of the FARs and HERs, respectively. These properties are stored in the IoT FND database. Table 3 describes referencing the information provided by the data model in tunnel provisioning templates.

Table 3 Data Model

Property	Description
far.eid	Returns the EID of the FAR. For example: `${far.eid}`
far.hostname	Returns the hostname of the FAR.
far.tunnelSrcInterface1	Returns the name of the FAR interface on which to establish the tunnel.

Table 3 Data Model (continued)

Property	Description
far.ipsecTunnelDestAddr1	Returns the name of the tunnel destination IP address on the HER.

far.ipv4Address(*clientId*, *linkAddress*, *userClass*)	Returns an IPv4 address. The IPv4 address method takes these parameters as input:
	■ clientId – DCHP Client Identifier for the DHCP request
	■ linkAddress – Link address for the DHCP request
	■ userClass – Value for the DHCP User Class option (defaults to "CG-NMS")
	To establish a loopback interface and assign it an address:
	`interface Loopback0`
	`ip address ${far.ipv4Address(dhcpClientId(far.enDuid, 0), far.dhcpV4LoopbackLink).address}/32`
	`ipv6 address ${far.ipv6Address(far.enDuid, 0, far.dhcpV6LoopbackLink).address}/128`
	`exit`

far.ipv4Subnet()	Returns a DHCP IPv4 subnet lease. This call takes a clientId and linkAddress as arguments. Construct the clientId from the FAR EID and interface ID number using the dhcpClientId() method provided in the template API. This method takes as input a DHCPv6 Identity Association Identifier (IAID) and a DHCP Unique Identifier (DUID) and generates the DHCPv4 client identifier, as specified in RFC 4361. This method provides consistency for how network elements are identified by the DHCP server. For example: `<#assign lease=far.ipv4Subnet(dhcpClientId(far.enDuid,iaId), far.dhcpV4TunnelLink)>`
far.[any device property]	Returns the value of the specified property. For example, far.tunnelSrcInterface1 returns the value of the FAR tunnelSrcInterface1 property.

far.interfaces(interfaceNamePrefix)	Returns a list of device interfaces that match the requested prefix (not case sensitive).
	Use square brackets to index list members, for example, [0], [1], [2], and so on. Use the <#list> construct to iterate list members.
	For example:
	`<#assign wanInterface = far.interfaces(interfaceNamePrefix)>`
	`  <#if (wanInterface[0].v4.addresses[0].address)??>`
	`...`

Addresses

Table 4 describes referencing addresses in the tunnel provisioning templates.

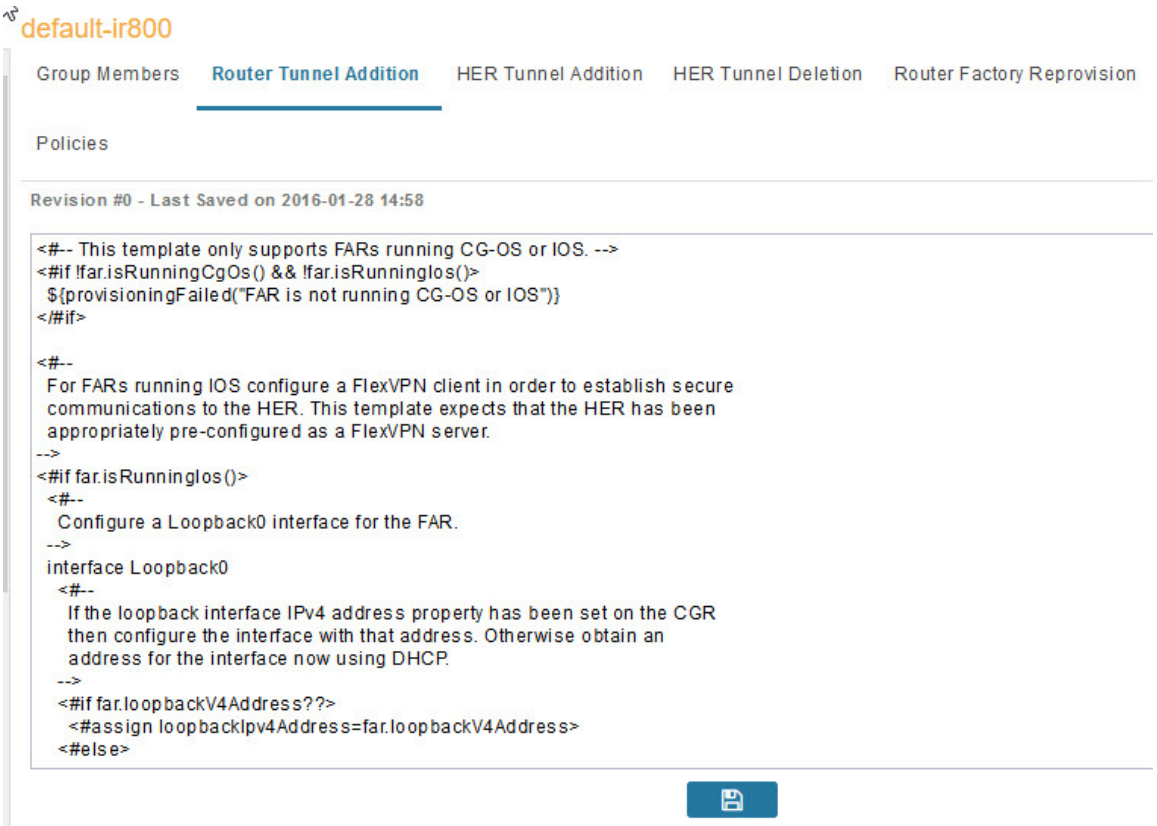
Table 4 Address References

Property	Description
address.address	Returns the address of the interface.
address.prefixLength	Returns the prefix length of the address.
address.prefix	Returns the address prefix.
address.subnetMask	Returns the subnet mask for the address.
address.wildcardMask	Returns the wildcard mask for the subnet.

Configuring the Field Area Router Tunnel Addition Template

To edit the FAR Tunnel Addition template to provide one end of an IPsec tunnel on FARs in the group:

1. Choose **CONFIG > Tunnel Provisioning**.
2. In the TUNNEL GROUPS pane, select the tunnel group with the template to edit.
3. Click the **Router Tunnel Addition** tab.



4. Modify the default template.

   Tip: Use a text editor to modify templates and copy the text into the template field in IoT FND.

5. Click the **Disk** icon to save changes.
6. Click **OK** to confirm the changes.

See also, Tunnel Provisioning Template Syntax.

Configuring the Head-End Router Tunnel Addition Template

Note: To ensure that both endpoints are in a matching subnet, this template must use the same IAID as the FAR template.

To edit the HER Tunnel Addition template to create the other end of the IPsec tunnel on HERs in the group:

1. Choose **CONFIG > Tunnel Provisioning**.
2. In the TUNNEL GROUPS pane, select a tunnel group.
3. Click the **HER Tunnel Addition** tab.
4. Modify the default HER addition template.
5. Click the **Disk** icon to save changes.
6. Click **OK** to confirm the changes.

Configuring the HER Tunnel Deletion Template

To edit the HER tunnel deletion template to delete existing tunnels to FARs at the other end of the tunnel:

1. Choose **CONFIG > Tunnel Provisioning**.
2. In the TUNNEL GROUPS pane, select the tunnel group whose template to edit.
3. Click the **HER Tunnel Deletion** tab.
4. Modify the default HER deletion template.
5. Click the **Disk** icon to save changes.
6. Click **OK** to confirm the changes.

Monitoring Tunnel Status

To view tunnel status, choose **OPERATIONS > Tunnel Status**. The Tunnel Status page lists devices and their provisioned tunnels and displays relevant information about tunnels and their status. Tunnels are provisioned between HERs and FARs.

When you select **Show Filter** at the top of the page (when selected, replaced by Hide Filter), a number of search fields appear. You can filter by all the Field Names listed in Table 5. The value entered in one search field will determine the available selections in the other fields. Select **Hide Filter** to remove the search fields.

Table 5 describes the tunnel status fields. To change the sort order of tunnels in the list by name, click the HER Name column heading. A small arrow next to the heading indicates the sort order.

Note: It takes time for the status of the newly created tunnel to be reflected in IoT FND.

Table 5 Tunnel Status Fields

Field	Description
HER Name	The EID of the HER at one end of the tunnel. To view the HER details, click its EID. **Note:** Because one HER can serve up to 500 FARs, there may be multiple tunnels in the list with the same HER EID. The Network Interfaces area of the Device Info page displays a list of tunnels configured on the HER. The Config Properties and Running Config tabs also contain information about tunnels configured on this HER.
HER Interface	The name of the HER tunnel interface. These names are automatically generated when tunnels are created (Tunnel1, Tunnel2, Tunnel3, and so on) or Virtual-Interface1, Virtual-Interface 2 and so on).
Admin Status	The administrative status of the tunnel (up or down). This indicates if the administrator enabled or disabled the tunnel.
Oper. Status	The operational status of the tunnel (up or down). If the tunnel is down, traffic does not flow through the tunnel, which indicates a problem to troubleshoot. Ping the HER and FAR to determine if they are online, or log on to the routers over SSH to determine the cause of the problem.
Protocol	The protocol used by the tunnel (IPSEC, PIM, or GRE).
HER Tunnel IP Address	The IP address of the tunnel at the HER side. Depending on the protocol used, the IP address appears in dotted decimal (IPv4) or hexadecimal (IPv6) slash notation.
HER IP Address	The destination IP address of the tunnel on the HER side.
FAR IP Address	The destination IP address of the tunnel on the FAR side.
FAR Interface	The name of the interface on the FAR used by the tunnel.
FAR Tunnel IP Address	The IP address of the tunnel on the FAR side. **Note:** The IP addresses on both sides of the tunnel are on the same subnet.
FAR Name	The EID of the FAR. To view the FAR details, click its EID. The Network Interfaces area of the Device Info page displays a list of tunnels configured on the FAR. The Config Properties and Running Config tabs also contain information about tunnels configured on this FAR.

Reprovisioning CGRs

In IoT FND, CGR reprovisioning is a process for modifying the configuration files on CGRs.

- CGR Reprovisioning Basics
- Tunnel Reprovisioning
- Factory Reprovisioning

Note: C800s do not support reprovisioning.

CGR Reprovisioning Basics

CGR Reprovisioning Actions

In IoT FND, you can perform the following two CGR reprovisioning actions at the Reprovisioning Actions pane of the Tunnel Provisioning page (**CONFIG > Tunnel Provisioning > Reprovisioning Actions**). You can also activate mesh firmware.

Tip: You can also type in the interface instead of selecting the preloaded interface values.

Reprovisioning Actions	Description
Factory Reprovisioning	Drop-down menu allows you to change the express-setup-config file loaded on the CGR during factory configuration. This file contains a minimal set of information and is loaded on the CGR at the factory. This file provides the CGR with information to contact IoT FND (call home) through the TPS Proxy after the CGR is deployed and powered on.
Tunnel Reprovisioning	Drop-down menu allows you to change the golden-config file on a CGR. This file has the tunnel configuration defined on the CGR.
Mesh Firmware Activation	Drop-down menu allows you to select the Interface (such as cellular, Ethernet, etc.) and Interface Type (IPv6 or IPv4).

Table 6 describes the fields on the Reprovisioning Actions pane.

Table 6 Reprovisioning Actions Pane Fields

Field	Description
Current Action	The current reprovisioning action being performed and the associated interface.
Reprovisioning Status	The status of the reprovisioning action.
Completed devices /All Scheduled Devices	The number of CGRs that were processed relative to the number of all CGRs scheduled to be processed.
Error devices/ All Scheduled Devices	The number of CGRs that reported an error relative to the number of all CGRs scheduled to be processed.
Name	The EID of the CGR.
Reprovisioning Status	The status of the reprovisioning action for this CGR.
Last Updated	The last time the status of the reprovisioning action for this CGR was updated.

Table 6 Reprovisioning Actions Pane Fields (continued)

Field	Description
Template Version	The version of the Field Area Router Factory Reprovision template being applied.
Error Message	The error message reported by the CGR, if any.
Error Details	The error details.

CGR Reprovisioning Sequence

When you start tunnel or factory reprovisioning on a tunnel provisioning group, the reprovisioning algorithm sequentially goes through 12 CGRs at a time and reprovisions them.

After IoT FND reprovisions a router successfully or if an error is reported, IoT FND starts the reprovisioning process for the next router in the group. IoT FND repeats the process until all CGRs are reprovisioned.

There is a timeout of 4 hours when reprovisioning each CGR in the group. If the CGR does not report successful reprovisioning or an error within the timeout period, then IoT FND changes the Reprovisioning Status of the CGR to Error and displays a timeout error and any further information displays in the Error Details field.

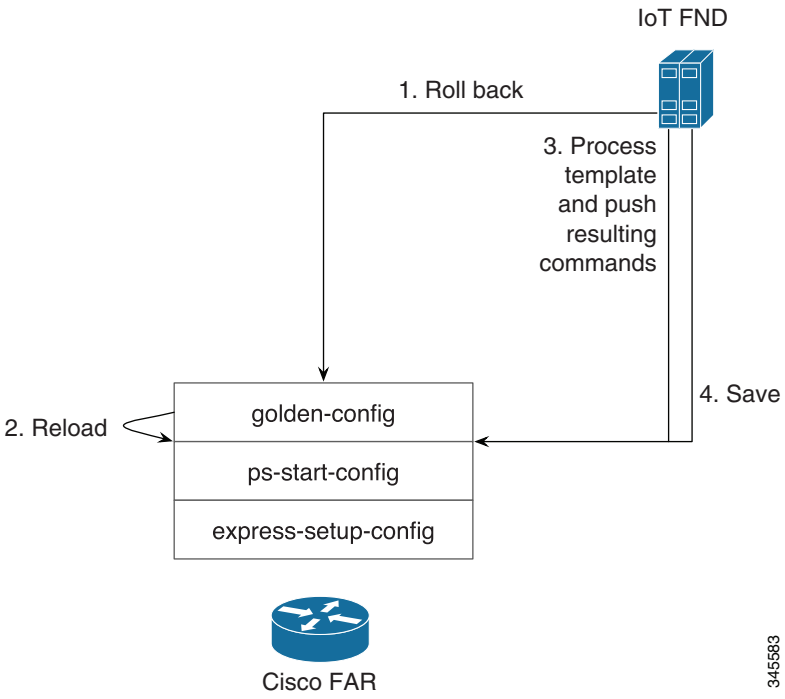
Tunnel Reprovisioning

If you make changes to the Field Area Router Tunnel Addition template and want all CGRs already connected to IoT FND reprovisioned with new tunnels based on the modified template, use the tunnel reprovisioning feature of IoT FND.

Tunnel reprovisioning places the CGR in a state where no tunnels are configured, and then initiates a new tunnel provisioning request. To reprovision tunnels, IoT FND sequentially goes through the FARs (12 at a time) in a tunnel provisioning group. For every CGR, IoT FND rolls back the configuration of the CGR to that defined in the ps-start-config template file.

After a rollback to ps-start-config, the CGR contacts IoT FND to request tunnel provisioning. IoT FND processes the Field Area Router Tunnel Addition template and sends the resultant configuration commands for creating new tunnels to the CGR. As shown in Figure 3, the tunnel provisioning process includes updating the golden-config file to include the new configuration information.

Figure 3 Tunnel Reprovisioning Process (CG-OS Routers)



Note: For CG-OS CGRs, FND initiates a config rollback and if the rollback fails, the router will go through a reload. Also, when IoT FND rolls back a CGR, IoT FND removes the corresponding tunnel information from the HERs to which the CGR was connected.

For Cisco IOS routers, the checkpoint files are before-tunnel-config, before-registration-config, and Express-setup-config. You perform a configuration replace for Cisco IOS based CGRs.

Note: The Field Area Router Factory Reprovision template is not used when performing tunnel reprovisioning.

To configure and trigger tunnel reprovisioning:

1. Choose **CONFIG > Tunnel Provisioning**.
2. In the TUNNEL GROUPS pane, select the tunnel group whose template to provision.
3. Click the **Reprovisioning Actions** tab.
4. From the Action drop-down menu, choose **Tunnel Reprovisioning**.
5. Click **Start**.

   IoT FND changes the Reprovisioning Status field to Initialized, and then to Running.

   Note: If you click **Stop** while tunnel reprovisioning is running, IoT FND stops the reprovisioning process only for the FARs in the queue that were not selected. However, for those CGRs in the queue that were selected for reprovisioning, the process completes (success or error) and cannot be stopped.

   The reprovisioning process completes after IoT FND finishes attempting to reprovision each CGR in the tunnel provisioning group. If a CGR cannot be reprovisioned, IoT FND displays the error message reported by the CGR.

Factory Reprovisioning

Use the Factory Reprovisioning feature in IoT FND to change the factory configuration of CGRs (express-setup-config).

Factory Reprovisioning involves these steps:

1. Sending the roll back command to the CGR.
2. Reloading the CGR.
3. Processing the Field Area Router Factory Reprovision template, and pushing the resultant commands to the CGR.
4. Saving the configuration in the express-setup-config file.

After these steps complete successfully, IoT FND processes the Field Area Router Tunnel Addition, Head-End Router Tunnel Addition, and Head-End Router Tunnel Deletion templates and pushes the resultant commands to the CGR (see Tunnel Provisioning Configuration Process).

To configure and trigger factory reprovisioning:

1. Choose **CONFIG > Tunnel Provisioning**.
2. In the TUNNEL GROUPS pane, select the tunnel group whose template you want to edit.
3. Click the **Router Factory Reprovision** tab and enter the template that contains the configuration commands to apply.

   Note: The Router Factory Reprovision template is processed twice during factory reprovisioning; once when pushing the configuration and again before saving the configuration in express-setup-config. Because of this, when making your own template, use the specific if/else condition model defined in the default template.

4. Click **Disk** icon to Save.
5. If needed, make the necessary modifications to the Field Area Router Tunnel Addition, Head-End Router Tunnel Addition, and Head-End Router Tunnel Deletion templates.
6. Click **Reprovisioning Actions** tab.
7. Select **Factory Reprovisioning**.

8. From the Interface drop-down menu, choose the CGR interface for IoT FND to use to contact the FARs for reprovisioning.
9. From the Interface Type drop-down menu, choose **IPv4** or **IPv6**.
10. Click the **Start** button.

    IoT FND changes the Reprovisioning Status field to Initialized, and then to Running.

    Note: If you click **Stop** while factory reprovisioning is running, IoT FND stops the reprovisioning process only for the FARs in the queue that were not selected. However, for those CGRs in the queue that were selected for reprovisioning, the process completes and cannot be stopped.

The reprovisioning process completes after IoT FND has finished attempting to reprovision each CGR in the tunnel provisioning group. If a CGR cannot be reprovisioned, IoT FND displays the error message reported by the CGR.

Sample Field Area Router Factory Reprovision Template

This sample template changes the WiFi SSID and passphrase in the factory configuration.

```
<#--IMPORTANT: This template is processed twice during factory reprovisioning. The if/else condition
described below is needed to determine which part of the template is applied.
In this example, if no schedule name wimaxMigrationRebootTimer is found in runningConfig, then the if
part of the if/else section is applied. During the second pass, this template runs the commands in the
else section and the no scheduler command is applied. If modifying this template, do not remove the
if/else condition or else the template fails. -->

<#if !far.runningConfig.text?contains("scheduler schedule name wimaxMigrationRebootTimer")>

<#--Comment: This is a sample of generating wifi ssid and passphrase randomly-->

wifi ssid ${far.randomSSID("PREFIX_")}
authentication key-management wpa2
wpa2-psk ascii ${far.randomPassword(10)}
exit

feature scheduler
scheduler job name wimaxMigration
reload
exit

scheduler schedule name wimaxMigrationRebootTimer
time start +02:00
job name wimaxMigration
exit

<#else>

no scheduler job name wimaxMigration
no scheduler schedule name wimaxMigrationRebootTimer

</#if>
```

Managing High Availability Installations

This section describes how to set up IoT FND for high availability, and includes the following sections:

CGR1240 High Availability Deployment Managed by IoT FND

Overview

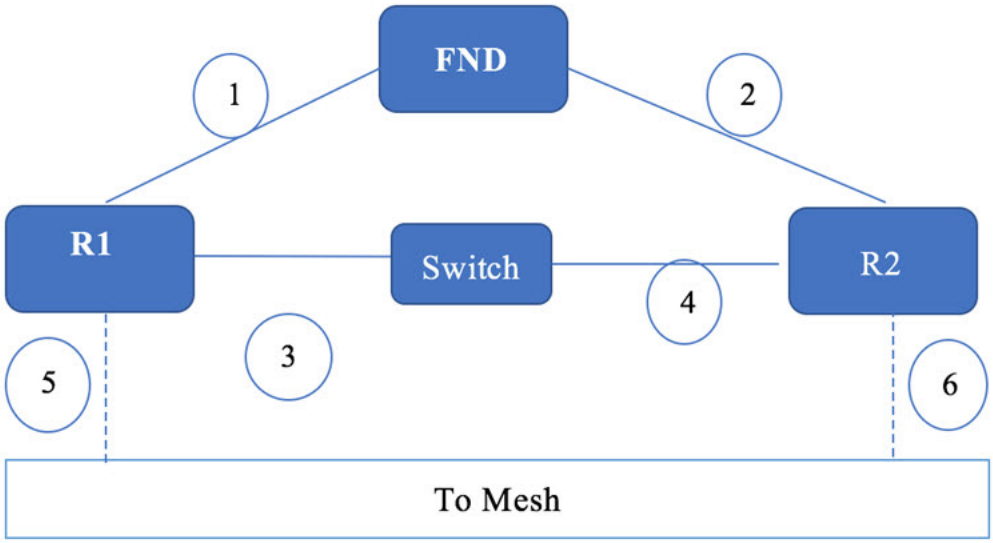
Note: This feature is only supported on CGR1240s and CGM-WPAN-OFDM modules.

To ensure connectivity to the mesh network, you can deploy the following CGR1240 and CGM-WPAN High Availability (HA) network. This network involves two CGR 1240s (R1 and R2), each installed with a CGM-WPAN-OFDM module, which are attached to a mesh network. A switch in the network provides access to the two CGM-WPAN-OFDM modules and manages their connection to the mesh network.

Key Facts:

- CGR1240 HA application is supported on NEW installs only. If you have an existing CGR1240 and CGM-WPAN-OFDM installation that you want to reconfigure for an HA deployment, you **must first disable** the components and then re-install them.
- You must have a minimum version of Cisco IOS 158-3.M installed on the two CGR1240s.
- Only one CGR1240 can be active at a time.
- Each CGR1240 has its own IP address.
- Each CGR1240 sends its HA state to FND. FND then updates its database.
- CGM-WPAN-OFDM modules must both be running the same software.

Figure 1 CGR1240 High Availability Deployment with CGM-WPAN-OFDM Modules



New Properties:

There are 3 new properties associated with the HA feature. All of the items listed below are populated in FND via CSV import **after** your two CGRs are installed and configured to support the HA deployment. See CGR Configuration to Support a HA Deployment.

- peerDevice (CGR1240): EIDs for the CGR1240 HA pairs represented as R1 and R2
- haTunnelip: IP address used by HSRP process
- ipsecTunnelDestAddr2: IP addresses of the HA destination tunnel

Table 1 Values You Must Assign To Support a Tunnel (Example Values Only)

peerDevice	haTunnelIp	ipsecTunnelDestAddr2
CGR1240/K9+FTX2150G04T	11.0.0.2	10.255.255.2
CGR1240/K9+FTX2150G04T	11.0.0.1	10.255.255.1

User Interface

IoT FND 4.3 has a new tab, WPAN HA, that appears on CGR1000 pages that displays details on the two CGRs (active and standby) and the HA status of each router.

The following items are tracked for each CGR1240 pair:

- Last Heard
- HA Status (Active or Standby)
- Peer Device
- Peer Status
- Peer HA Status

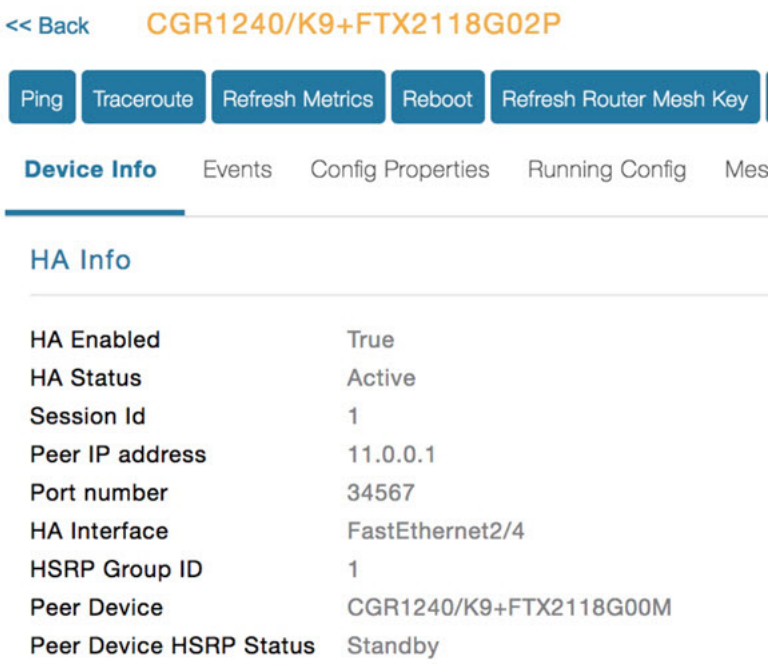
Group ID

Overall HA Status

You can also view additional information for CGR HA pairs at the DEVICE > FIELD DEVICES page for the CGR1000:

- Mesh Link Keys (Key Refresh Time and Key Expiration Time)
- HA Info on Device Info tab (Figure 2): Enabled state, HA Status, Session ID, Peer IP address, Port Number, HA Interface, HSRP Group ID, Peer Device, Peer Device HSRP Status

Figure 2 HA Info



CGR Configuration to Support a HA Deployment

Listed below are sample configurations you would configure on the two CGRs in a HA deployment:

CGR Router 1 (R1)-Active

```
track 1 interface fa2/4 line-protocol
track 2 interface wpan 4/1 line-protocol
Conf t
Int fa2/4
ip address 11.0.0.2 255.255.255.128
standby version 2
standby 12 ip 11.0.0.11
standby 12 preempt
standby 12 track 1 decrement 10
standby 12 track 2 decrement 10
standby 12 track 3 decrement 10
duplex auto
speed auto
interface Wpan4/1
```

```
bandwidth inherit
no ip address
ip broadcast-address 0.0.0.0
standby 12 track 1 decrement 10
standby 12 track 2 decrement 10
no ip route-cache
ieee154 beacon-async min-interval 10 max-interval 10 suppression-coefficient 1
ieee154 dwell window 12400 max-dwell 400
ieee154 panid 440
ieee154 ssid hatest
outage-server 2002:1111:2222::250
peer-to-peer
rpl dag-lifetime 60
rpl dio-dbl 4
rpl dio-min 14
ha enable
      control-interface fa2/4 standby 12
      peer-ip 11.0.0.1
authentication host-mode multi-auth
authentication port-control auto
ipv6 address 2002:DB8:1111:2222::1/64
ipv6 dhcp server iok-dhcpd6 rapid-commit
dot1x pae authenticator
```

CGR Router 2 (R2)-Standby

```
track 1 interface fa2/4 line-protocol
track 2 interface wpan 4/1 line-protocol
interface fa2/4
ip address 11.0.0.1 255.255.255.0
standby version 2
standby 12 ip 11.0.0.11
standby 12 preempt
standby 12 track 1 decrement 10
standby 12 track 2 decrement 10
standby 12 track 3 decrement 10
duplex auto
speed auto
interface Wpan4/1
bandwidth inherit
no ip address
ip broadcast-address 0.0.0.0
standby 12 track 1 decrement 10
standby 12 track 2 decrement 10
no ip route-cache
shutdown
ieee154 beacon-async min-interval 10 max-interval 10 suppression-coefficient 1
ieee154 dwell window 12400 max-dwell 400
ieee154 panid 440
ieee154 ssid hatest
outage-server 2002:1111:2222::250
peer-to-peer
frame-counter
rpl dag-lifetime 60
rpl dio-dbl 4
rpl dio-min 14
ha enable
      control-interface fa2/4 standby 12
      peer-ip 11.0.0.2
authentication host-mode multi-auth
authentication port-control auto
ipv6 address 2002:DB8:1111:2222::1/64
ipv6 dhcp server iok-dhcpd6 rapid-commit
dot1x pae authenticator
```

Overview of IoT FND High Availability

This section provides an overview of IoT FND high availability installations, including the following sections:

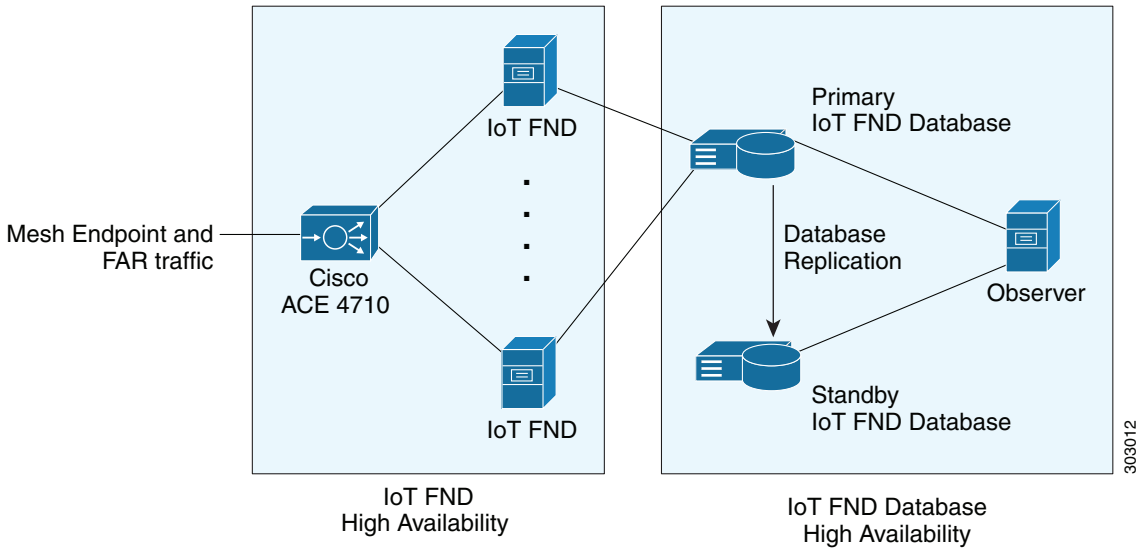
- Load Balancer
- Server Heartbeats
- Database High Availability
- Tunnel Redundancy

IoT FND is a critical application for monitoring and managing a connected grid. IoT FND High Availability (IoT FND HA) solutions address the overall availability of IoT FND during software, network, or hardware failures.

IoT FND provides two main levels of HA, as shown in Figure 3:

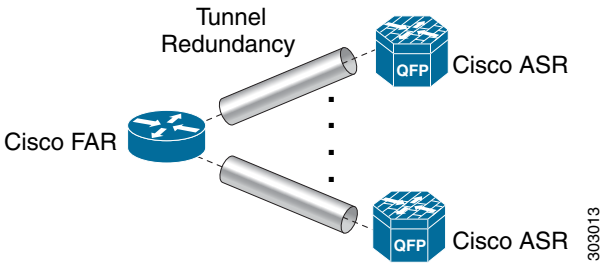
- IoT FND Server HA—This is achieved by connecting multiple IoT FND servers to a Cisco ACE 4710 load balancer. Traffic originating at MEs, FARs, and ASRs goes to the load balancer, which uses a round-robin protocol to distribute the load among the IoT FND cluster servers.
- IoT FND Database HA—This is achieved by configuring two IoT FND Database servers: a primary server and a standby (or secondary) server. When the primary database receives new data it sends a copy to the standby database. A separate system runs the Observer (the Observer can also run on the standby server), which is a program that monitors the IoT FND Database servers. If the primary database fails, the Observer configures the standby server as the new primary database. IoT FND Database HA works in single and cluster IoT FND server deployments.

Figure 3 IoT FND Server and Database HA



In addition to IoT FND Server and Database HA, IoT FND improves reliability by adding tunnel redundancy. This is achieved by defining multiple tunnels between one FAR and multiple ASRs. If one tunnel fails, the FAR routes traffic through another tunnel.

Figure 4 IoT FND Tunnel Redundancy



IoT FND HA addresses these failure scenarios:

Failure Type	Description
IoT FND server failure	If a server within a IoT FND server cluster fails, the load balancer routes traffic to the other servers in the cluster.
IoT FND database failures	If the primary database fails, the associated standby database becomes the primary database. This is transparent to the IoT FND servers. All IoT FND servers in the cluster connect to the new primary database.
Tunnel failure	If a tunnel fails, traffic flows through another tunnel.

Load Balancer

The Load Balancer (LB) plays a critical role in IoT FND HA, as it performs these tasks:

- Load balances traffic destined for IoT FND.
- Maintains heartbeats with servers in the cluster and detects any failure. If a IoT FND server fails, the LB directs traffic to other cluster members.

Cisco recommends using the Cisco ACE 4710 (Cisco ACE) as the load balancer in this deployment. See http://www.cisco.com/en/US/partner/products/ps7027/tsd_products_support_series_home.html for information on the Cisco ACE 4710.

Server Heartbeats

The LB maintains heartbeats with each IoT FND server in the cluster. In the health monitoring mechanism adopted by the IoT FND solution (there are alternate solutions), the heartbeats are regular GET messages to IoT FND on port 80. IoT FND expects an HTTP 200 OK response from an active IoT FND server.

You can configure these heartbeat parameters on the LB:

- Periodicity of probes—This is the number of seconds between heartbeats. The default value on the Cisco ACE is 15 seconds
- Number of retries—This is the number of times the LB tries to send a heartbeat to a non-responding IoT FND server before declaring it down. The default number of retries is 3.
- Regular checks after failure detection—The LB checks whether the server is back online at this time interval. The default failure detection check value is 60 seconds.

Database High Availability

IoT FND Database HA works in IoT FND single-server and cluster deployments. IoT FND HA uses Oracle Active Dataguard to deploy Oracle HA. To configure HA for the IoT FND Database, use the Oracle Recovery Manager (RMAN) and Dataguard Management CLI (DGMGRL).

The IoT FND Database HA configuration process involves:

- Configuring the primary and secondary databases the same on separate physical servers.

  Note: The secondary database server is also referred to as the standby database.

  Note: There is a possibility of losing some data during a database failover.

- Configuring data replication to be performed over SSL using an Oracle *wallet*. The wallet contains a self-signed certificate to facilitate quick deployment.

  Note: The Oracle wallet bundled with the IoT FND RPMs uses self-signed certificates. You can configure custom certificates and wallet to facilitate replication.

  Note: There is no performance impact when performing data replication over SSL.

- Using the sys user for replication and *not* cgms_dev.

- Configuring replication as asynchronous to prevent performance bottlenecks.

By default, IoT FND connects to the database using TCP over port 1522. Replication uses TCPS (TCP over SSL) on port 1622.

The scripts for configuring IoT FND Database HA are included in the IoT FND Oracle Database RPM package (cgms-oracle-*version_number*.x86_64.rpm). When you install the IoT FND Database, the HA scripts are located in $ORACLE_HOME/cgms/scripts/ha.

Tunnel Redundancy

To add another layer of redundancy to your IoT FND deployment, configure multiple tunnels to connect every FAR in a FAR tunnel provisioning group to multiple ASRs. For example, you could configure IoT FND to provision two tunnels for every FAR. One tunnel is active over the Cellular interface, while the redundant tunnel is configured to communicate with a second ASR over the WiMAX interfaces.

To configure tunnel redundancy, you need to:

1. Add ASRs to a tunnel provisioning group.
2. Modify the tunnel provisioning templates to include commands to create additional tunnels.
3. Define policies that determine the mapping between interfaces on the FAR and ASR interfaces:
   - Configuring Tunnel Provisioning Policies
   - Modifying the Tunnel Provisioning Templates for Tunnel Redundancy

HA Guidelines and Limitations

Note the following about IoT FND HA configurations:

- IoT FND HA does not include HA support for other network components like FARs, ASRs, and the load balancer.
- Zero service downtime is targeted by IoT FND HA, but it is not guaranteed.
- All IoT FND nodes must on the same subnet.
- All IoT FND nodes must run on similar hardware.
- All IoT FND nodes must run the same software version.

- Run the IoT FND setup script (/opt/cgms/bin/setupCgms.sh) on all the nodes.
- Run the DB migration script (/opt/cgms/bin/db-migrate) on only one node.
- The /opt/cgms/bin/print_cluster_view.sh script displays information about IoT FND cluster members.

HA Installation for FND

Be sure to enter the following information in the /opt/cgms/bin/cgms.conf file:

To fetch mesh keys from the primary CGR in a HA setup, enter these parameters:

- cgr-ha-fetch-mesh-key-attempts = 3 <-- you can modify the number of attempts to fetch the mesh keys
- cgr-ha-fetch-mesh-key-delay-mins = 1 <-- number of minutes (interval) between mesh-key-attempts
- CLUSTER_BIND_ADDR= a.b.c.d
- UDP_MULTICAST_ADDR= w.x.y.z

where CLUSTER_BIND_ADDR is the IP address of the server itself and UDP_MULTICAST_ADDR must be the same on all instances. It can be either an IPv4 multicast address or an IPv6 address which is not used in the network.

Configuring IoT FND Installations for HA

This section describes the various configuration settings for IoT FND HA installations, including the following sections:

- Setting Up IoT FND Database for HA
- Disabling IoT FND Database HA
- Load-Balancing Policies
- Running LB Configuration Example
- Configuring Tunnel Provisioning Policies
- Modifying the Tunnel Provisioning Templates for Tunnel Redundancy

Setting Up IoT FND Database for HA

To set up the IoT FND Database HA:

1. Set up the standby database (see Setting Up the Standby Database).

   Note: Always configure the standby database first.

   - The default SID for the standby server is **cgms_s** and *not* cgms.
   - Before setting up the standby server for HA, ensure that the environment variable $ORACLE_SID on the standby server is set to **cgms_s**.
   - The port is always 1522.

2. Set up the primary database (see Setting Up the Primary Database).
   - The default SID for the primary server is **cgms**.
   - Before setting up the primary server for HA, ensure that the environment variable $ORACLE_SID on the primary server is set to **cgms**.
3. Set up IoT FND for database HA (see Setting Up IoT FND for Database HA).
4. Set up the database Observer (see Setting Up the Observer).

Setting Up the Standby Database

To set up the standby database server for HA, run the setupStandbyDb.sh script. This script prompts for configuration information needed for the standby database, including the IP address of the primary database.

```
$ ./setupStandbyDb.sh
$ Are you sure you want to setup a standby database ? (y/n)? y

09-20-2012 13:59:18 PDT: INFO: User response: y
09-20-2012 13:59:18 PDT: INFO: CGMS_S database does not exist.
Enter the SYS DBA password. NOTE: This password should be same as the one set on the primary server:
Re-enter password for SYS DBA:
09-20-2012 13:59:58 PDT: INFO: User entered SYS DBA password.

Enter new password for CG-NMS database:
Re-enter new password CG-NMS database:
09-20-2012 14:00:09 PDT: INFO: User entered CG-NMS DB password.
Enter primary database server IP address: 192.168.1.12
09-20-2012 14:00:27 PDT: INFO: Cleaning up instance - cgms_s
DGMGRL for Linux: Version 11.2.0.3.0 - 64bit Production
…
Total System Global Area  329895936 bytes
Fixed Size    2228024 bytes
Variable Size  255852744 bytes
Database Buffers   67108864 bytes
Redo Buffers    4706304 bytes
...
09-20-2012 14:00:29 PDT: INFO: ========== CGMS_S Database Setup Completed Successfully ==========
```

Setting Up the Primary Database

To set up the primary database server for HA, run the setupHaForPrimary.sh script. This script prompts for configuration information needed for the primary database, including the IP address of the standby database.

```
$ ./setupHaForPrimary.sh
[oracle@pdb ha]$ ./setupHaForPrimary.sh
09-20-2012 13:58:39 PDT: INFO: ORACLE_BASE: /home/oracle/app/oracle
09-20-2012 13:58:39 PDT: INFO: ORACLE_HOME: /home/oracle/app/oracle/product/11.2.0/dbhome_1
09-20-2012 13:58:39 PDT: INFO: ORACLE_SID : cgms
09-20-2012 13:58:39 PDT: INFO: Make sure the above environment variables are what you expect

Are you sure you wish to configure high availability for this database server ? (y/n)? y

09-20-2012 13:58:45 PDT: INFO: User response: y
Enter standby database server IP address: 192.168.1.10
09-20-2012 13:58:56 PDT: INFO: Secondary listener reachable. Moving on with configuration
mkdir: cannot create directory `/home/oracle/app/oracle/oradata/cgms': File exists
09-20-2012 13:58:58 PDT: INFO: Reloading the listener to pick the new settings
```

```
LSNRCTL for Linux: Version 11.2.0.3.0 - Production on 20-SEP-2012 13:58:58

…
DGMGRL> 09-20-2012 14:14:54 PDT: INFO: Please start the 'Observer' on appropriate server for ha
monitoring
Total time taken to perform the operation: 975 seconds
09-20-2012 14:14:54 PDT: INFO: ==========  Completed Successfully ==========
```

Setting Up the Observer

The Observer should run on a separate server, but can be set up on the server hosting the standby database.

Note: The password required for running Observer is the same as the SYS DBA password. See *Cisco IoT Field Network Director Installation Guide- Oracle Only Deployment, Release 4.3.x,* "Creating the IoT FND Oracle Database."

To set up the Observer:

1. On a separate server, run the observer script.

```
$ ./manageObserver.sh start cgms_s password
$ DGMGRL for Linux: Version 11.2.0.3.0 - 64bit Production
...
Welcome to DGMGRL, type "help" for information.
DGMGRL> Connected.
DGMGRL> Observer started
```

2. Run the getHaStatus.sh script to verify that the database is set up for HA.

```
$ ./getHaStatus.sh
...
Configuration - cgms_dgconfig

  Protection Mode: MaxPerformance
  Databases:
    cgms   - Primary database
    cgms_s - (*) Physical standby database

Fast-Start Failover: ENABLED

Configuration Status:
SUCCESS

DGMGRL>
Database - cgms

  Role:          PRIMARY
  Intended State:  TRANSPORT-ON
  Instance(s):
    cgms

Database Status:
SUCCESS

DGMGRL>
Database - cgms_s

  Role:          PHYSICAL STANDBY
  Intended State:  APPLY-ON
  Transport Lag:   0 seconds
  Apply Lag:       0 seconds
  Real Time Query: OFF
  Instance(s):
    cgms_s
```

```
Database Status:
SUCCESS
```

Setting Up IoT FND for Database HA

To set up IoT FND for database HA:

1. Stop IoT FND.
2. Run the setupCgms.sh script.

   The script prompts you to change the database settings. Enter **y**. Then, the script prompts you to enter the primary database server information (IP address, port, and database SID). After that, the script prompts you to add another database server. Enter **y**. Then, the script prompts you to enter the standby database server information (IP address, port, and database SID), as follows:

   Note: IoT FND always uses port 1522 to communicate with the database. Port 1622 is only used by the database for replication.

```
# cd /opt/cgms/bin
# ./setupCgms.sh
09-13-2012 17:10:00 PDT: INFO: ========== CG-NMS Setup Started - 2012-09-13-17-10-00 ==========
09-13-2012 17:10:00 PDT: INFO: Log file: /opt/cgms/bin/../server/cgms/log/cgms_setup.log

Are you sure you want to setup CG-NMS (y/n)? y

09-13-2012 17:10:02 PDT: INFO: User response: y

Do you want to change the database settings (y/n)? y

09-13-2012 17:10:05 PDT: INFO: User response: y

Enter database server IP address [128.107.154.246]: 128.107.154.246
09-13-2012 17:11:02 PDT: INFO: Database server IP: 128.107.154.246

Enter database server port [1522]:
09-13-2012 17:11:07 PDT: INFO: Database server port: 1522

Enter database SID [cgms]:
09-13-2012 17:11:12 PDT: INFO: Database SID: cgms

Do you wish to configure another database server for this CG-NMS ? (y/n)? y

09-13-2012 17:11:18 PDT: INFO: User response: y
Enter database server IP address []: 128.107.154.20
09-13-2012 17:11:02 PDT: INFO: Database server IP: 128.107.154.20
Enter database server port []: 1522
09-13-2012 17:11:07 PDT: INFO: Database server port: 1522
Enter database SID []: cgms_s
09-13-2012 17:11:12 PDT: INFO: Database SID: cgms_s
09-13-2012 17:11:18 PDT: INFO: Configuring database settings. This may take a while. Please wait
...
09-13-2012 17:11:19 PDT: INFO: Database settings configured.

Do you want to change the database password (y/n)? y

09-13-2012 17:15:07 PDT: INFO: User response: y

Enter database password:
Re-enter database password:
```

```
09-13-2012 17:15:31 PDT: INFO: Configuring database password. This may take a while. Please wait
...
09-13-2012 17:15:34 PDT: INFO: Database password configured.

Do you want to change the keystore password (y/n)? n

09-13-2012 17:16:18 PDT: INFO: User response: n

Do you want to change the web application 'root' user password (y/n)? n

09-13-2012 17:16:34 PDT: INFO: User response: n

Do you want to change the FTP settings (y/n)? n

09-13-2012 17:16:45 PDT: INFO: User response: n
09-13-2012 17:16:45 PDT: INFO: ========== CG-NMS Setup Completed Successfully ==========
```

Disabling IoT FND Database HA

To disable IoT FND Database HA:

1. On the server running the Observer program, stop the Observer:

```
$ ./manageObserver.sh stop cgms_s password
DGMGRL for Linux: Version 11.2.0.3.0 - 64bit Production

Copyright (c) 2000, 2009, Oracle. All rights reserved.

Welcome to DGMGRL, type "help" for information.
DGMGRL> Connected.
DGMGRL> Done.
$ Observer stopped
```

2. On the standby IoT FND Database server, delete the standby database:

```
$ ./deleteStandbyDb.sh

Are you sure you want to delete the standby database ? All replicated data will be lost (y/n)? y

09-20-2012 14:27:02 PDT: INFO: User response: y
09-20-2012 14:27:02 PDT: INFO: Cleaning up instance - cgms_s
DGMGRL for Linux: Version 11.2.0.3.0 - 64bit Production

Copyright (c) 2000, 2009, Oracle. All rights reserved.

Welcome to DGMGRL, type "help" for information.
DGMGRL> Connected.
DGMGRL> Done.
DGMGRL> DGMGRL for Linux: Version 11.2.0.3.0 - 64bit Production

Copyright (c) 2000, 2009, Oracle. All rights reserved.

Welcome to DGMGRL, type "help" for information.
DGMGRL> Connected.
DGMGRL> Disabled.
DGMGRL> 09-20-2012 14:27:06 PDT: INFO: Removing dataguard configuration
DGMGRL for Linux: Version 11.2.0.3.0 - 64bit Production

Copyright (c) 2000, 2009, Oracle. All rights reserved.

Welcome to DGMGRL, type "help" for information.
DGMGRL> Connected.
DGMGRL> Removed configuration
DGMGRL> 09-20-2012 14:27:07 PDT: INFO: Stopping the database
```

```
SQL*Plus: Release 11.2.0.3.0 Production on Thu Sep 20 14:27:07 2012

Copyright (c) 1982, 2011, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.3.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> ORA-01109: database not open


Database dismounted.
ORACLE instance shut down.
SQL> Disconnected from Oracle Database 11g Enterprise Edition Release 11.2.0.3.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

LSNRCTL for Linux: Version 11.2.0.3.0 - Production on 20-SEP-2012 14:27:19

Copyright (c) 1991, 2011, Oracle.  All rights reserved.

Connecting to
(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=test-scale-15krpm)(PORT=1522))(CONNECT_DATA=(SERVER=DEDIC
ATED)(SERVICE_NAME=cgms_s)))
The command completed successfully
Cleaning up instance - cgms_s
09-20-2012 14:27:29 PDT: INFO: ==========  Completed Successfully ==========
```

3. On the primary IoT FND Database server, delete the HA configuration:

```
$ ./deletePrimaryDbHa.sh
Are you sure you want to delete the high availability configuration ? All replicated data will be
lost (y/n)? y

09-20-2012 14:25:25 PDT: INFO: User response: y
09-20-2012 14:25:25 PDT: INFO: Removing secondary configuration from primary

SQL*Plus: Release 11.2.0.3.0 Production on Thu Sep 20 14:25:25 2012

Copyright (c) 1982, 2011, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.3.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL>
System altered.
...
SQL> Disconnected from Oracle Database 11g Enterprise Edition Release 11.2.0.3.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options
09-20-2012 14:25:28 PDT: INFO: Removing data guard config files
09-20-2012 14:25:28 PDT: INFO: Removing standby redo logs
09-20-2012 14:25:29 PDT: INFO: Creating listener file
09-20-2012 14:25:29 PDT: INFO: Listener successfully configured.
09-20-2012 14:25:29 PDT: INFO: Recreating tnsnames ora file
09-20-2012 14:25:29 PDT: INFO: reloading the listener

LSNRCTL for Linux: Version 11.2.0.3.0 - Production on 20-SEP-2012 14:25:29

Copyright (c) 1991, 2011, Oracle.  All rights reserved.
```

```
Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=test-scale-15krpm-db2)(PORT=1522)))
The command completed successfully

LSNRCTL for Linux: Version 11.2.0.3.0 - Production on 20-SEP-2012 14:25:30

Copyright (c) 1991, 2011, Oracle.  All rights reserved.

Starting /home/oracle/app/oracle/product/11.2.0/dbhome_1/bin/tnslsnr: please wait...

TNSLSNR for Linux: Version 11.2.0.3.0 - Production
System parameter file is /home/oracle/app/oracle/product/11.2.0/dbhome_1/network/admin/listener.ora
Log messages written to
/home/oracle/app/oracle/diag/tnslsnr/test-scale-15krpm-db2/cgmstns/alert/log.xml
Listening on: (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=test-scale-15krpm-db2)(PORT=1522)))

Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=test-scale-15krpm-db2)(PORT=1522)))
STATUS of the LISTENER
------------------------
Alias                     cgmstns
Version                   TNSLSNR for Linux: Version 11.2.0.3.0 - Production
Start Date                20-SEP-2012 14:25:30
Uptime                    0 days 0 hr. 0 min. 0 sec
Trace Level               off
Security                  ON: Local OS Authentication
SNMP                      OFF
Listener Parameter File
/home/oracle/app/oracle/product/11.2.0/dbhome_1/network/admin/listener.ora
Listener Log File
/home/oracle/app/oracle/diag/tnslsnr/test-scale-15krpm-db2/cgmstns/alert/log.xml
Listening Endpoints Summary...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=test-scale-15krpm-db2)(PORT=1522)))
Services Summary...
Service "cgms" has 1 instance(s).
  Instance "cgms", status UNKNOWN, has 1 handler(s) for this service...
The command completed successfully
09-20-2012 14:25:30 PDT: INFO: ==========  Completed Successfully ==========
```

Load-Balancing Policies

Table 2 describes the load-balancing policy for each type of traffic the LB supports:

Table 2

Traffic	Load Balancing Policy
HTTPS traffic to and from browsers and IoT FND API clients (IPv4; ports 80 and 443)	The LB uses Layer 7 load balancing for all traffic from Web browsers and IoT FND API clients. The LB uses stickiness for general HTTPS traffic.
For FAR IPv4 traffic going to ports 9121 and 9120: ■ Tunnel Provisioning on port 9120 over HTTPS ■ Regular registration and periodic on 9121 over HTTPS	The LB uses Layer 3 load balancing for all FAR traffic. This is the traffic from the FAR to IoT FND.
For IPv6 CSMP traffic to and from mesh endpoints (MEs): ■ UDP traffic over port 61624 – Registration – Periodic transmission of metrics – Firmware push – Configuration push ■ UDP traffic over port 61625 For outage notifications sent by MEs.	The LB uses Layer 3 load balancing for all ME traffic to port 61624, and outage messages to port 61625.

Running LB Configuration Example

The following is an example of the running configuration of a properly configured IoT FND LB:

```
# show running-config
Generating configuration....

ssh maxsessions 10

boot system image:c4710ace-t1k9-mz.A5_1_1.bin

hostname cgnmslb2
interface gigabitEthernet 1/1
  switchport access vlan 10
  no shutdown
interface gigabitEthernet 1/2
  description server-side
  switchport access vlan 11
  no shutdown
interface gigabitEthernet 1/3
  description client-side
  switchport access vlan 8
  no shutdown
interface gigabitEthernet 1/4
  switchport access vlan 55
  no shutdown
```

```
access-list ALL line 8 extended permit ip any any
access-list everyone line 8 extended permit ip any any
access-list everyone line 16 extended permit icmp any any
access-list ipv6_acl line 8 extended permit ip anyv6 anyv6
access-list ipv6_acl2 line 8 extended permit icmpv6 anyv6 anyv6



ip domain-lookup
ip domain-name cisco.com
ip name-server 171.68.226.120
ip name-server 171.70.168.183

probe http probe_cgnms-http
  port 80
  interval 15
  passdetect interval 60
  expect status 200 200
  open 1

rserver host 12-12-1-31
  ip address 12.12.1.31
  conn-limit max 4000000 min 4000000
  probe probe_cgnms-http
  inservice
rserver host 12-12-1-32
  ip address 12.12.1.32
  conn-limit max 4000000 min 4000000
  probe probe_cgnms-http
  inservice
rserver host 2002-cafe-server-202
  description realserver 2002:cafe:server::202
  ip address 2002::202
  conn-limit max 4000000 min 4000000
  probe probe_cgnms-http
  inservice
rserver host 2002-cafe-server-211
  ip address 2002:cafe:server::211
  conn-limit max 4000000 min 4000000
  probe probe_cgnms-http
  inservice

serverfarm host cgnms_2
  description cgnms-serverfarm
  probe probe_cgnms-http
  rserver 2002-cafe-server-202 61624
    conn-limit max 4000000 min 4000000
    inservice
  rserver 2002-cafe-server-211 61624
    conn-limit max 4000000 min 4000000
    inservice
serverfarm host cgnms_2_ipv4
  probe probe_cgnms-http
  rserver 12-12-1-31
    conn-limit max 4000000 min 4000000
    inservice
  rserver 12-12-1-32
    conn-limit max 4000000 min 4000000
    inservice

sticky ip-netmask 255.255.255.255 address source CGNMS_SRC_STICKY
  serverfarm cgnms_2_ipv4
```

```
class-map type management match-any remote_access
  2 match protocol xml-https any
  3 match protocol icmp any
  4 match protocol telnet any
  5 match protocol ssh any
  6 match protocol http any
  7 match protocol https any
  8 match protocol snmp any
class-map type management match-all ssh_allow_access
  2 match protocol ssh any
class-map match-any virtual-server-cgnms
  2 match virtual-address 2002:server:cafe::210 udp eq 61624
class-map match-any vs_cgnms_ipv4
  3 match virtual-address 12.12.1.101 tcp eq https
  4 match virtual-address 12.12.1.101 tcp eq 9120
  5 match virtual-address 12.12.1.101 tcp eq 9121
  6 match virtual-address 12.12.1.101 tcp eq 8443
  7 match virtual-address 12.12.1.101 tcp any

policy-map type management first-match remote_mgmt_allow_policy
  class remote_access
    permit

policy-map type loadbalance first-match virtual_cgnms_l7
  class class-default
    serverfarm cgnms_2
policy-map type loadbalance first-match vs_cgnms_l7_v4
  class class-default
    sticky-serverfarm CGNMS_SRC_STICKY

policy-map multi-match cgnms_policy_ipv6
  class virtual-server-cgnms
    loadbalance vip inservice
    loadbalance policy virtual_cgnms_l7
    loadbalance vip icmp-reply active
policy-map multi-match int1000
  class vs_cgnms_ipv4
    loadbalance vip inservice
    loadbalance policy vs_cgnms_l7_v4
    loadbalance vip icmp-reply active

interface vlan 8
  bridge-group 1
  access-group input everyone
  access-group input ipv6_acl
  no shutdown
interface vlan 10
  bridge-group 2
  access-group input everyone
  access-group input ipv6_acl
  service-policy input int1000
  no shutdown
interface vlan 11
  bridge-group 2
  access-group input everyone
  access-group input ipv6_acl
  no shutdown
interface vlan 55
  bridge-group 1
  access-group input everyone
  access-group input ipv6_acl
  service-policy input cgnms_policy_ipv6
```

```
  no shutdown

interface bvi 1
  ipv6 enable
  ip address 2002:server:cafe::206/64
  no shutdown
interface bvi 2
  ip address 12.12.1.100 255.255.255.0
  no shutdown

domain cisco.com

ip route 2011::/16 2002:server:cafe::101
ip route 2001:server:cafe::/64 2002:cafe::101
ip route 11.1.0.0 255.255.0.0 12.12.1.33
ip route 15.1.0.0 255.255.0.0 12.12.1.33
ip route 13.211.0.0 255.255.0.0 12.12.1.33

context VC_Setup1
  allocate-interface vlan 40
  allocate-interface vlan 50
  allocate-interface vlan 1000

username admin password 5 $1$CB34uAB9$BW8a3ijjxvBGttuGtTcST/  role Admin domain
default-domain
username www password 5 $1$q/YDKDp4$9PkZl1SBMQW7yZ7E.sOZA/  role Admin domain de
fault-domain

ssh key rsa 1024 force
```

Configuring Tunnel Provisioning Policies

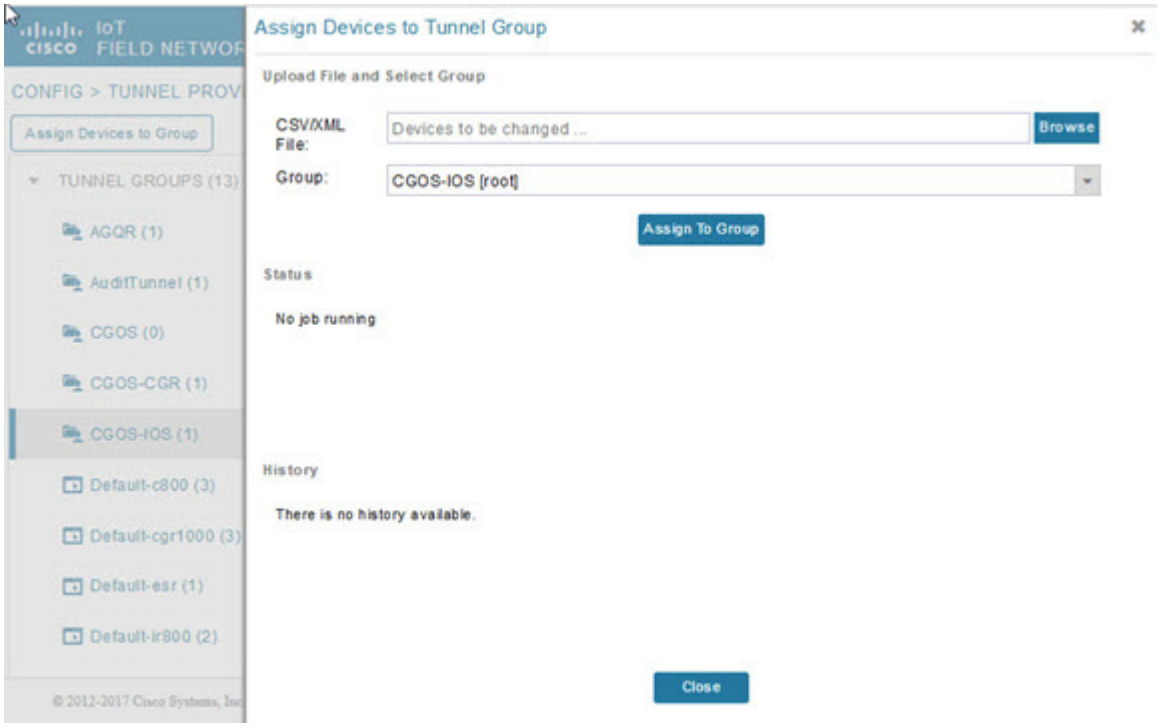
Use tunnel policies to configure multiple tunnels for a FAR. Each tunnel is associated with an interface on a FAR and an HER. If a tunnel provisioning group has one or more HERs, IoT FND displays a policy in the Tunnel Provisioning Policies tab (**Config > Tunnel Provisioning**). Use this policy to configure FAR-to-HER interface mapping.

To map FAR-to-HER interfaces in IoT FND:

1. Choose **CONFIG > Tunnel Provisioning**.
2. In the TUNNEL GROUPS pane, select a group to configure with tunnel redundancy.
3. Create a CSV or XML file that lists the HERs to add to the group in the format *EID, device type*, as follows:

   ```
   eid,deviceType
   asr-0, asr1000
   asr-1, asr1000
   asr-2, asr1000
   ```

4. Click **Assign Devices to Group** to import the file and add HERs to the group.



Note: A HER can be a member of multiple tunnel provisioning groups.

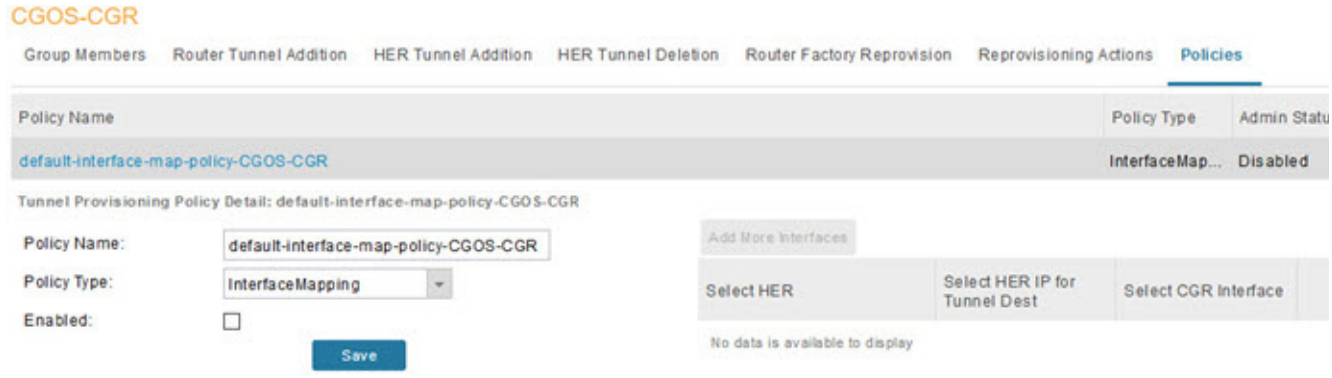
5. With the tunnel provisioning group selected, click the **Policies** tab.

   By default, IoT FND displays the **default-interface-mapping-policy-*tunnel-group*** name for the selected tunnel group within the Policy Name panel.

   Note: Interface-mapping is the only policy type currently supported in IoT FND.

   IoT FND displays one interface mapping entry for every HER in the group. You can add or remove interface mapping entries as needed.

6. Click the Policy Name link within the Policy Name Panel to open an entry panel. In the Policy Name field, enter the name of the policy.



To **add** an interface-mapping entry to the policy, click **Add More Interfaces** (button found above Select HER listing right-side of page). To **delete** an entry, click **Delete** (**X**) for that entry.

7. To configure an interface-mapping entry, click the Policy Name link, and complete the following as necessary:
   a. To select a different HER, click the currently selected HER and choose a different one from the **Select a HER** drop-down menu.
   b. To select the HER IP for the tunnel destination on the HER, click the selected interface and choose a different one from the **Select HER IP** drop-down menu.
   c. To select the FAR interface that maps to the selected HER interface, choose an interface from the **Select CGR Interface** drop-down menu.
   d. Click **Update**.
8. To enable the policy, check the **Enabled** check box.
9. Click **Save**.

Modifying the Tunnel Provisioning Templates for Tunnel Redundancy

After defining the tunnel provisioning policy for a tunnel provisioning group, modify the Field Area Router Tunnel Addition and the Head-End Router Tunnel Addition templates to include commands to establish the multiple tunnels defined in the policy.

EXAMPLE: Field Area Router Tunnel Addition Template

In this example, bold text indicates the changes made to the default Field Area Router Tunnel Addition template to create multiple tunnels:

```
<#--
  Configure a Loopback0 interface for the FAR. This is done first as features
  look for this interface and use it as a source.

  This is independent of policies
-->
interface Loopback0
  <#--
    Now obtain an IPv4 address that can be used to for this FAR's Loopback
    interface. The template API provides methods for requesting a lease from
    a DHCP server. The IPv4 address method requires a DHCP client ID and a link
    address to send in the DHCP request. The 3rd parameter is optional and
    defaults to "CG-NMS". This value is sent in the DHCP user class option.
    The API also provides the method "dhcpClientId". This method takes a DHCPv6
    Identity association identifier (IAID) and a DHCP Unique  IDentifier (DUID)
    and generates a DHCPv4 client identifier as specified in RFC 4361. This
    provides some consistency in how network elements are identified by the
    DHCP server.
  -->
  ip address ${far.ipv4Address(dhcpClientId(far.enDuid, 0), far.dhcpV4LoopbackLink).address}/32
  <#--
    Now obtain an IPv6 address that can be used to for this FAR's loopback
    interface. The method is similar to the one used for IPv4, except clients
    in DHCPv6 are directly identified by their DUID and IAID. IAIDs used for
    IPv4 are separate from IAIDs used for IPv6, so we can use zero for both
    requests.
  -->
  ipv6 address ${far.ipv6Address(far.enDuid, 0, far.dhcpV6LoopbackLink).address}/128
exit

<#-- Make certain the required features are enabled on the FAR. -->
feature crypto ike
feature ospf
feature ospfv3
```

```
feature tunnel
<#-- Features ike and tunnel must be enabled before ipsec. -->
feature crypto ipsec virtual-tunnel

<#--
  Toggle on/off the c1222r feature to be certain it uses the Loopback0
  interface as its source IP.
-->
no feature c1222r
feature c1222r

<#-- Configure Open Shortest Path First routing processes for IPv4 and IPv6. -->
router ospf 1
exit
router ospfv3 2
exit

<#--
  Now that OSPF has been configured complete the configuration of Loopback0.
-->
interface Loopback0
  ip router ospf 1 area ${far.ospfArea1!"1"}
  ipv6 router ospfv3 2 area ${far.ospfV3Area1!"0"}
exit

<#-- Configure Internet Key Exchange for use by the IPsec tunnel(s). -->
crypto ike domain ipsec
  identity hostname
  policy 1
    <#-- Use RSA signatures for the authentication method. -->
    authentication rsa-sig
    <#-- Use the 1536-bit modular exponential group. -->
    group 5
  exit
exit
crypto ipsec transform-set IPSecTransformSet esp-aes 128 esp-sha1-hmac
crypto ipsec profile IPSecProfile
  set transform-set IPSecTransformSet
exit

<#--
  Define template variables to keep track of the next available IAID (IPv4)
  and the next available tunnel interface number. We used zero when leasing
  addresses for Loopback0, so start the IAID at one.
-->
<#assign iaId = 1>
<#assign interfaceNumber = 0>

<#--
  The same logic is needed for each of the IPsec tunnels, so a macro is used
  to avoid duplicating configuration. The first parameter is the prefix to
  use when looking for the WAN interface on the FAR to use for the source of
  the tunnel. The second parameter is the OSPF cost to assign to the tunnel.
-->
<#macro configureTunnel interfaceNamePrefix destinationInterface her tunnelIndex ospfCost>
  <#--
    If an interface exists on the FAR whose name starts with the given prefix
    and an IPv4 address as been assigned to that interface then the IPsec
    tunnel can be configured, otherwise no tunnel will be configured. The
    template API interfaces method will return all interfaces whose name
    starts with the given prefix.
  -->
  <#assign wanInterface = far.interfaces(interfaceNamePrefix)>
```

```
  <#-- Check if an interface was found and it has an IPv4 address. -->
  <#if (wanInterface[0].v4.addresses[0].address)??>
    <#--
      Determine the HER destination address to use when configuring the tunnel.
      If the optional property "ipsecTunnelDestAddr1" has been set on this FAR
      then use the value of that property. Otherwise look for that same property
      on the HER. If the property is not set on the FAR or the HER, then fallback
      to using an address on the HER GigabitEthernet0/0/0 interface.
    -->
    <#assign destinationAddress = her.interfaces(destinationInterface)[0].v4.addresses[0].address>

    <#if !(destinationAddress??)>
      ${provisioningFailed("Unable to determine the destination address for IPsec tunnels")}
    </#if>
    interface Tunnel${interfaceNumber}
      <#assign interfaceNumber = interfaceNumber + 1>
      description IPsec tunnel to ${her.eid}
      <#--
        For a tunnel interface two addresses in their own tiny subnet are
        needed. The template API provides an ipv4Subnet method for leasing an
        IPv4 from a DHCP server. The parameters match those of ipv4Address,
        with a fourth optional parameter that can be used to specify the
        prefix length of the subnet to request. If not specified the prefix
        length requested will default to 31, which provides the two addresses
        needed for a point to point link.

        NOTE: If the DHCP server being used does not support leasing an IPv4
        subnet, then this call will have to be changed to use the ipv4Address
        method and the DHCP server will have to be configured to respond
        appropriately to the request made here and the second request that
        will have to be made when configuring the HER side of the tunnel.
        That may require configuring the DHCP server with reserved addresses
        for the client identifiers used in the calls.
      -->
      <#assign lease = far.ipv4Subnet(dhcpClientId(far.enDuid, tunnelIndex), far.dhcpV4TunnelLink)>
      <#assign iaId = iaId + 1>
      <#-- Use the second address in the subnet for this side of the tunnel. -->
      ip address ${lease.secondAddress}/${lease.prefixLength}
      ip ospf cost ${ospfCost}
      ip ospf mtu-ignore
      ip router ospf 1 area ${far.ospfArea1!"1"}
      tunnel destination ${destinationAddress}
      tunnel mode ipsec ipv4
      tunnel protection ipsec profile IPSecProfile
      tunnel source ${wanInterface[0].name}
      no shutdown
    exit
  </#if>
</#macro>

<#--
  Since we are doing policies for each tunnel here, the list of policies passed to this template can be
  iterated over to get the tunnel configuration viz interface mapping

  tunnelObject.ipSecTunnelDestInterface is the "interface on CGR"
  tunnelObject.ipSecTunnelSrcInterface is the "interface on HER"
  tunnelObject.her is the HER of interest
 -->

<#list far.tunnels("ipSec") as tunnelObject>
    <@configureTunnel tunnelObject.ipSecTunnelDestInterface tunnelObject.ipSecTunnelSrcInterface
tunnelObject.her tunnelObject.tunnelIndex 100/>  <----- Loop through policies (aka Tunnels)
</#list>

<#--
```

```
  Make certain provisioning fails if we were unable to configure any IPsec
  tunnels. For example this could happen if the interface properties are
  set incorrectly.
-->
<#if iaId = 1>
  ${provisioningFailed("Did not find any WAN interfaces to use as the source for IPsec tunnels")}
</#if>

<#--
  Configure an IPv6-in-IPv4 GRE tunnel to allow IPv6 traffic to reach the data
  center.
-->
<#macro configureGreTunnel destinationInterface her tunnelIndex>

<#assign destinationAddress = her.interfaces(destinationInterface)[0].v4.addresses[0].address>

<#if !(destinationAddress??)>
     ${provisioningFailed("Unable to determine the destination address for GRE tunnels")}
</#if>

interface Tunnel${interfaceNumber}
  <#assign interfaceNumber = interfaceNumber + 1>
  description GRE IPv6 tunnel to ${her.eid}
  <#--
    The ipv6Subnet method is similar to the ipv4Subnet method except instead
    of obtaining an IPv4 subnet it uses DHCPv6 prefix delegation to obtain an
    IPv6 prefix. The prefix length will default to 127, providing the two
    addresses needed for the point to point link. For the IAID, zero was used
    when requesting an IPv6 address for loopback0, so use one in this request.
  -->
  <#assign lease = far.ipv6Subnet(far.enDuid, tunnelIndex, far.dhcpV6TunnelLink)>
  ipv6 address ${lease.secondAddress}/${lease.prefixLength}
  ipv6 router ospfv3 2 area ${far.ospfV3Area1!"0"}
  ospfv3 mtu-ignore
  tunnel destination ${destinationAddress}
  tunnel mode gre ip
  tunnel source Loopback0
  no shutdown
exit

</#macro>

<#-- Loop through the policies for GRE tunnels -->
<#list far.tunnels("gre") as greTunnelObj>
    <@configureGreTunnel greTunnelObj.greDestInterface greTunnelObj.her greTunnelObj.tunnelIndex/>
</#list>
```

Head-End Router Tunnel Addition Template

In this example, bold text indicates the changes made to the default Head-End Router Tunnel Addition template to create multiple tunnels:

```
<#--
  Define template variables to keep track of the IAID (IPv4) that was used by
  the FAR template when configuring the other end of the tunnel. This template
  must use the same IAID in order to locate the same subnet that was leased by
  the FAR template so both endpoints are in the matching subnet.
-->
<#assign iaId = 1>

<#--
```

```
  The same logic is needed for each of the IPsec tunnels, so a macro is used.
-->
<#macro configureTunnel ipSecTunnelSrcInterface ipSecTunnelDestInterface her tunnelIndex ospfCost>
  <#--
    Only configure the HER tunnel end point if the FAR tunnel end point was
    configured. This must match the corresponding logic in the FAR tunnel
    template. The tunnel will not have been configured if the WAN interface
    does not exist on the FAR or does not have an address assigned to it.
  -->
  <#assign wanInterface = far.interfaces(ipSecTunnelDestInterface)>
  <#if (wanInterface[0].v4.addresses[0].address)??>
    <#-- Obtain the full interface name based on the prefix. -->
    <#assign interfaceName = wanInterface[0].name>
    <#--
      Locate a tunnel interface on the HER that is not in use. The template
      API provides an unusedInterfaceNumber method for this purpose. All of
      the parameters are optional. The first parameter is a name prefix
      identifying the type of interfaces, it defaults to "tunnel". The second
      parameter is a lower bound on the range the unused interface number must
      be in, it defaults to zero. The third parameter is the upper bound on
      the range, it defaults to max integer (signed). The method remembers
      the unused interface numbers it has returned while the template is
      being processed and excludes previously returned numbers. If no unused
      interface number meets the constraints an exception will be thrown.
    -->
    interface Tunnel${her.unusedInterfaceNumber()}
      description IPsec tunnel to ${far.eid}
      <#assign lease = far.ipv4Subnet(dhcpClientId(far.enDuid, tunnelIndex), far.dhcpV4TunnelLink)>
      <#assign iaId = iaId + 1>
      ip address ${lease.firstAddress} ${lease.subnetMask}
      ip ospf cost ${ospfCost}
      ip ospf mtu-ignore
      tunnel destination ${wanInterface[0].v4.addresses[0].address}
      tunnel mode ipsec ipv4
      tunnel protection ipsec profile IPSecProfile
      tunnel source ${ipSecTunnelSrcInterface}
      no shutdown
    exit
    router ospf 1
      network ${lease.prefix} ${lease.wildcardMask} area ${far.ospfArea1!"1"}
    exit
  </#if>
</#macro>

<#list far.tunnels("ipSec") as tunnelObject>
        <@configureTunnel tunnelObject.ipSecTunnelSrcInterface tunnelObject.ipSecTunnelDestInterface
tunnelObject.her tunnelObject.tunnelIndex 100/>
</#list>

<#--
  Configure an IPv6-in-IPv4 GRE tunnel to allow IPv6 traffic to reach the data
  center.
-->
<#macro configureGreTunnel greSrcInterface her tunnelIndex>
interface Tunnel${her.unusedInterfaceNumber()}
  description GRE IPv6 tunnel to ${far.eid}
  <#assign lease = far.ipv6Subnet(far.enDuid, tunnelIndex, far.dhcpV6TunnelLink)>
  ipv6 address ${lease.firstAddress}/${lease.prefixLength}
  ipv6 enable
  ipv6 ospf 2 area ${far.ospfV3Area1!"0"}
  ipv6 ospf mtu-ignore
  tunnel destination ${far.interfaces("Loopback0")[0].v4.addresses[0].address}
  tunnel mode gre ip
  tunnel source ${greSrcInterface}
exit
```

```
</#macro>

<#-- Loop through the policies for GRE tunnels -->
<#list far.tunnels("gre") as greTunnelObj>
        <@configureGreTunnel greTunnelObj.greSrcInterface greTunnelObj.her greTunnelObj.tunnelIndex/>
</#list>
```