HP Network Node Manager

for the Windows, HP-UX, Solaris, and Linux operating systems

Software Version: 7.53

Using Extended Topology

Manufacturing Part Number: T4929-90006

Document Release Date: February 2008

Software Release Date: February 2008

Legal Notices

Warranty

Restricted Rights Legend

Copyright Notices

Trademark Notices

Support

You can visit the HP Software support web site at:

www.hp.com/go/hpsoftwaresupport

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online software support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require an active support contract. To find more information about support access levels, go to the following URL:

http://h20230.www2.hp.com/new_access_levels.jsp

To register for an HP Passport ID, go to the following URL:

http://h20229.www2.hp.com/passport-registration.html

Contents

1 Introduction to Extended Topology Functionality

Extended Topology Functionality

HP OpenView Network Node Manager Advanced Edition's Extended Topology functionality (Extended Topology) discovers layer 2 device information and displays device connectivity. HP OpenView Network Node Manager Advanced Edition (NNM) is a software application that discovers and monitors your IP network.

Extended Topology discovers and displays additional device connectivity information for devices that are discovered by NNM. Extended Topology collects this information from specific MIBs contained in discovered devices.

Detailed layer 2 network information that is discovered by Extended Topology is presented in several Dynamic Views. Dynamic Views describes the family of browser-based views whose content is created as a result of choices you make when you launch a view. The views provide the most current status information available.

Extended Topology also discovers and monitors network domains that use overlapping private IP addresses. See Chapter 9, Overlapping Address Domains for more information.

Obtain More Information About Extended Topology

To view information from the online help, select the **?** icon in the upper right corner of any Dynamic View. The online help contains a great deal of information to help you interact with Dynamic Views.

To view the NNM Release Notes, select **Help** → **NNM** → **Release Notes**.

NNM Basics

NNM Environment Variables

NNM manuals use directory path names that are independent of the underlying file structure of the operating system you are using. For each NNM directory, a single path name is used rather than multiple path names that vary by system. For example, the path name for the NNM log file is stated as follows:

- *UNIX*:

  `$OV_LOG rather than /var/opt/OV/share/log`

- *Windows*:

  `%OV_LOG% rather than \install_dir\log\`

The *Using Extended Topology* manual uses the same path name convention as the NNM manuals and refers to universal path names throughout the manual. For example, the path name for the setupExtTopo.ovpl file is stated as follows:

- *UNIX*:

  `$OV_BIN rather than /opt/OV/bin`

- *Windows*:

  `%OV_BIN% rather than \install_dir\bin\`

You can set up universal path names using environment variables. See *Managing Your Network with HP OpenView Network Node Manager* for details on environment variables and how to set them up for your environment.

Stop and Restart Processes

At times, it is necessary to stop and restart the NNM processes. To stop and restart the processes, execute the following command:

UNIX:

To stop all the processes, run the following command:

```
$OV_BIN/ovstop -c
```

To start all the processes, run the following command:

```
$OV_BIN/ovstart -c
```

Windows:

To stop all the processes, run the following command:

```
%OV_BIN%\ovstop -c
```

To start all the processes, run the following command:

```
%OV_BIN%\ovstart -c
```

➤ You may want to stop, start, or check the status of a **single process**. To do this, use the following commands:

To stop a process, run the following command:

```
ovstop -c <process name>
```

To start a process, run the following command:

```
ovstart -c <process name>
```

To check the status of a process, run the following command:

```
ovstatus -c <process name>
```

2 Extended Topology Discovery and Configuration

Basics of Extended Topology Discovery

Extended Topology discovery is different from NNM discovery. Rather than incrementally discovering new nodes over time, you configure Extended Topology to periodically discover network information. See Extended Topology Discovery Process on page 41 for more information.

In contrast, when you first install NNM and start the NNM background processes, all IP and layer 2 devices on your network are automatically discovered and mapped out. The discovery and mapping process may take several hours to discover all the devices on your network. Over time, NNM continues to discover new nodes.

After you install NNM and Extended Topology, you must enable Extended Topology before the initial Extended Topology discovery can start. In a very large environment, Extended Topology discovery can take up to several hours. If you complete a good NNM discovery before you enable Extended Topology, you discovery time can be reduced. See *Managing Your Network with HP OpenView Network Node Manager* for more information about NNM discovery.

In larger environments, Extended Topology divides your IPv4 network into discovery zones to reduce the amount of time to complete an Extended Topology discovery. See Extended Topology Discovery on page 22 for more information about configuring zones.

➤ Extended Topology collects detailed information from devices whose existence was discovered by NNM. Extended Topology collects this information from specific MIBs contained in discovered devices.

Extended Topology discovers the following information and protocols:

- Layer 2 Information

  Extended Topology discovers layer 2 connection information and uses this information to map managed nodes and their neighboring port relationships. The result is a more accurate view of your network.

- VLAN Information

  Extended Topology discovers and displays VLAN information from managed devices.

- Board and Port Information on Cisco Devices

  Extended Topology discovers and displays Cisco board and port information for devices that support the CISCO-C2900-MIB, CISCO-STACK-MIB, or CISCO-RHINO-MIB. This includes the board, interface, and address information contained in supported Cisco devices.

- Multiple Links Between Devices

  When multiple links between two devices behave as one logical link, that configuration is said to have a *trunk* or *port aggregation*, depending on the vendor and technology used. Extended Topology discovers and displays this information in two ways:

  — For Cisco devices, an *aggregated-ports* symbol (circle with internal lines) displays.

  — For other devices, a *redundant links* symbol (circle with internal diamond) displays.

- Multi-Link Trunking and Split Multi-Level Trunking

  Multi-Link Trunking (MLT) for Nortel devices is supported. With MLT, multiple ports logically function as a single port with aggregated bandwidth.

  The Split Multi-Link Trunking (SMLT) Nortel architecture is also supported. SMLT provides a way to reduce single points of failure by creating multiple paths from switches to the network core. SMLT prevents loops in your network and provides architecture that helps make your network resilient.

- ATM Information

  Extended Topology discovers ATM information such as Virtual Path Identifier (VPI) and Virtual Channel Identifier (VCI).

- VRRP Information

  The Virtual Router Redundancy Protocol (VRRP) protocol is supported. In a static routing environment, VRRP eliminates single points of failure. VRRP dynamically assigns responsibility for virtual routers to one of many VRRP routers on a LAN. This protocol currently supports only IPv4 routers.

- Overlapping Address Domain Information

  Extended Topology can monitor multiple network domains that contain overlapping addresses from the private internet address space. This situation occurs when you manage multiple private IP address domains from outside of their gateways, and some of the addresses overlap.

Extended Topology Information and Distribution

Extended Topology only discovers information from locally managed nodes and does not pass Extended Topology information from a collection station to a management station. To open Dynamic Views that include information from the Extended Topology, you need to point your web browsers to an object's primary collection station.

Extended Topology Discovery

After you install Extended Topology, you must run the following script to enable Extended Topology and start your first Extended Topology discovery:

- *UNIX*:

```
$OV_BIN/setupExtTopo.ovpl
```

- *Windows*:

```
%OV_BIN%\setupExtTopo.ovpl
```

This script initializes Extended Topology and does the following:

- Determines the protocols your environment supports and decides what information to discover
- Checks the system kernel parameters and tells you if you need to make any system modifications
- Asks if you want to discover information about certain protocols
- Asks you to set up your initial administrator user name and password for Dynamic Views

  You can change the default user name and password or accept the default values. See the *Extended Topology and APA Deployment Guide* in the whitepaper directory for more information about user authentication.

- Determines the number of nodes that are managed by NNM
- In larger environments, the script can divide the IPv4 devices discovered by NNM into smaller discovery zones if you so choose. See Automatic Zone Configuration on page 31 for more information.

Zone-based discovery consumes fewer computer resources and results in the potential for a *much* faster network discovery. Zones can be thought of as “islands of connectivity.” They are discovered independently and are later brought together through their edge connections.

When possible, Extended Topology immediately proceeds with its first discovery without recommending zone configuration.

Start Extended Topology Discovery

The `setupExtTopo.ovpl` script starts your first discovery.

If you want to manually restart your Extended Topology discovery, you can select **Initiate Full Discovery Now** from the Extended Topology Configuration page. See Open the Extended Topology Configuration Page on page 43 for more information.

You can also use the following script to restart Extended Topology discovery:

- *UNIX*:

```
$OV_BIN/etrestart.ovpl
```

- *Windows*:

```
%OV_BIN%\etrestart.ovpl
```

➤ If you want the `etrestart.ovpl` script to interrupt a discovery in progress or another `etrestart.ovpl` script that was executed earlier, use `etrestart.ovpl -force`.

See the *etrestart.ovpl* reference page for more information. Details about how to access reference pages are in the online help.

Use Zones for Discovery

You can discover your environment as a whole, or you can break up your discovered environment into zones. Using zones improves discovery performance and allows you to focus your discovery on specific areas where change has occurred.

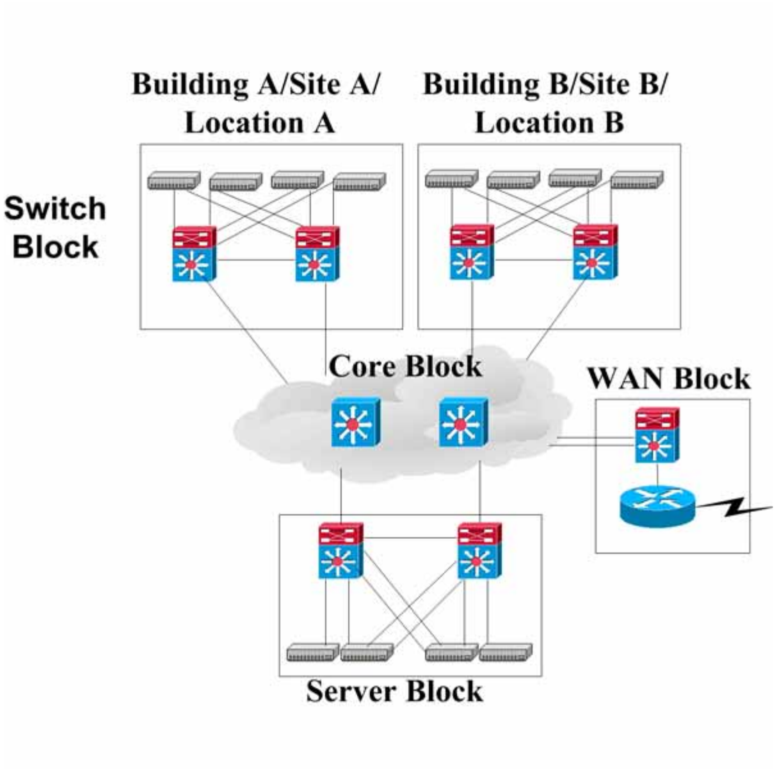
Common Zone Configuration Examples

Network designers deploy various enterprise network designs to meet their customers' requirements. The network designs described in this chapter contain patterns that can be used to plan and configure the Extended Topology discovery.

Campus Environment Scenario

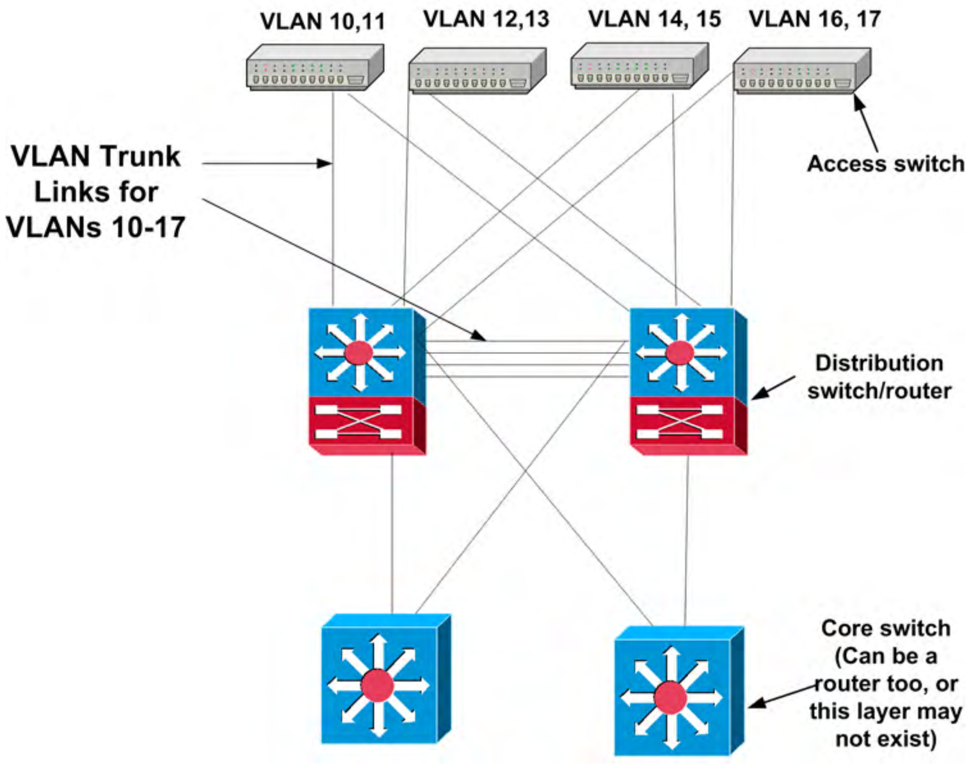
Suppose you are managing the modern campus environment shown in Figure 1

Figure 1 A Typical Campus Environment



Each switch block from Figure 1 connects to the core block as shown in Figure 2

Figure 2 Core Block


VLAN 10,11
VLAN 12,13
VLAN 14, 15
VLAN 16, 17
VLAN Trunk Links for VLANs 10-17
Access switch
Distribution switch/router
Core switch (Can be a router too, or this layer may not exist)

The VLAN numbering shown in Figure 2 is only a sample. The access switches can, and probably will, have overlapping VLAN IDs. Assume that you create DNS names for devices based upon geography for this enterprise network (for example `switch.bldg4.atlanta.company.com`). Figure 2 shows a redundant network to better demonstrate how to configure Extended Topology zones for more efficient and accurate discovery.

The rest of this chapter discusses zone discovery configurations that refer to the network models shown in Figure 1. You can interchange the terms *buildings*, *locations*, or *sites* depending on the size of your company.

The following strategies can be replicated for all campuses of the enterprise. The following discussion does not go into any detail for WAN block designs, since Extended Topology is designed to discover layer 2 and other protocol information. It does not discover WAN information.

General Guidelines

Do not separate the following equipment into different zones:

1. Directly connected switches
2. Switches connected to routers
3. Switches in the same VLAN (that is, having the same global VLAN names or IDs)

Separating the above equipment into separate zones will result in inaccurate discoveries. You can place a router in multiple zones to avoid separating it.

Extended Topology discovers router-to-router connectivity information only if the routers support a discovery protocol such as CDP, EDP, FDP, and so on. If your routers do not support one of these discovery protocols, it does not matter how you divide them up because Extended Topology cannot obtain connectivity information from them.

If you do not need to view router connectivity within the Extended Topology neighbor view, configure the core routers in as few zones as possible.

If the network size in each building is too small, combine the networks of two or more buildings to get a bigger zone. Add the core routers for each building into that zone.

Based on OpenView tests, you may want to start with zones of 200 nodes or less. The OpenView tests discovered devices with port densities of 24, and were run on HP C3000 and Sun Ultra 60 workstations with 512 MB of RAM. However, in some environments successful discoveries can have up to four times that many devices in a zone. You might want to experiment to decide what zone size works best for you, based on the following parameters:

- The port densities of the devices you plan to discover.
- Extended Topology system memory size.
- Extended Topology system CPU size.
- The number of VLANs your network contains.
- Other resource settings.

Remember to use the `[Test Zone Configuration]` button to test your zones.

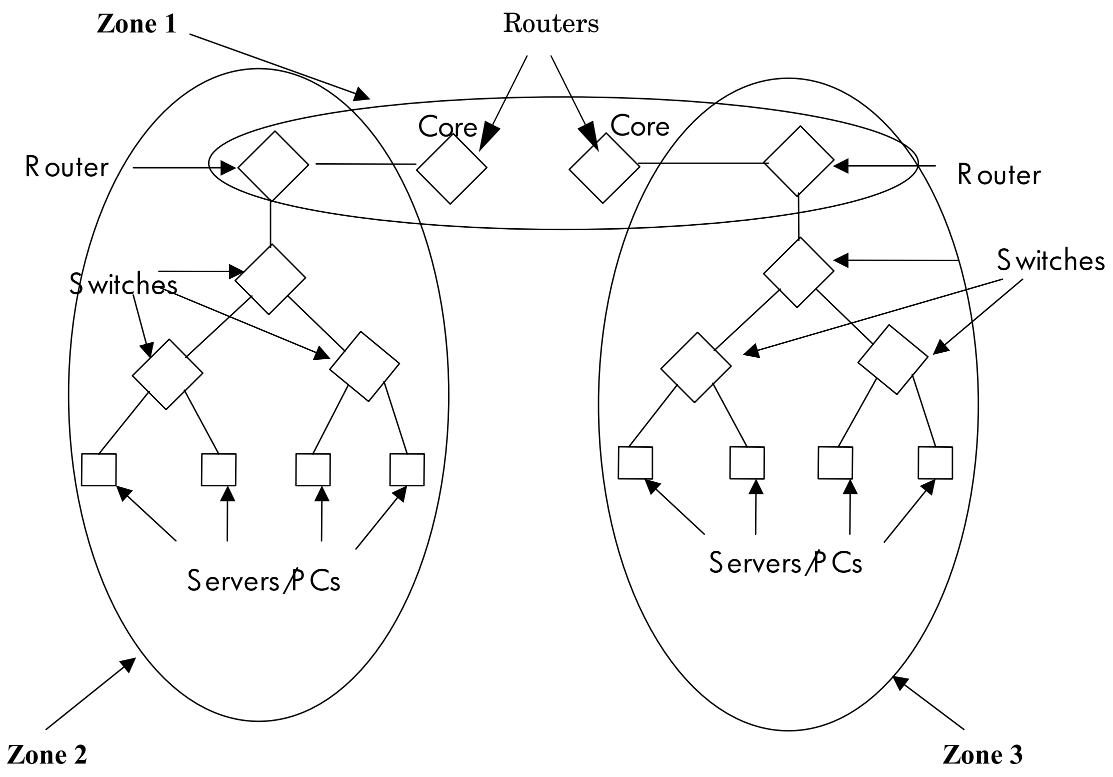
Specific Guidelines

These guidelines are for some sample network types that may be commonly deployed.

Possibility 1: Core Devices are Routers

- The network contains core devices that are all routers.
- Campus buildings/sites are mostly switched, with router interfaces placed between switch blocks or VLANs.

Figure 3 Core is Routed



Suggested Zone Configuration

- To configure discovery for the switch-to-switch and switch-to-router connectivity, do the following:

  Refer to Figure 3 for this discussion. Put all of the access switches, distribution switches/routers, and core router(s) that service the building into zone 1 using the name wildcard method for defining a zone. Using this method, you can create a zone for every building in your campus, provided it meets the above criteria.

- To configure discovery for the router-to-router connectivity, do the following:

  Refer to Figure 3 for this discussion. Configure the core routers into one zone by themselves. Splitting the core routers into multiple zones may increase management traffic from bordering routers to the core routers for each zone you add the core routers to. You should split large core routers into a few zones to speed up discovery. Doing this will speed up discovery at the cost of higher management traffic. It is a design choice that you need to make at zone configuration time.

Possibility 2: Core Devices are Switches

The network contains core devices that are all switches.

Campus buildings or sites are mostly switched, with router interfaces placed between switch blocks and/or VLANs.

Figure 4 Core is Switched: Using Two Zones

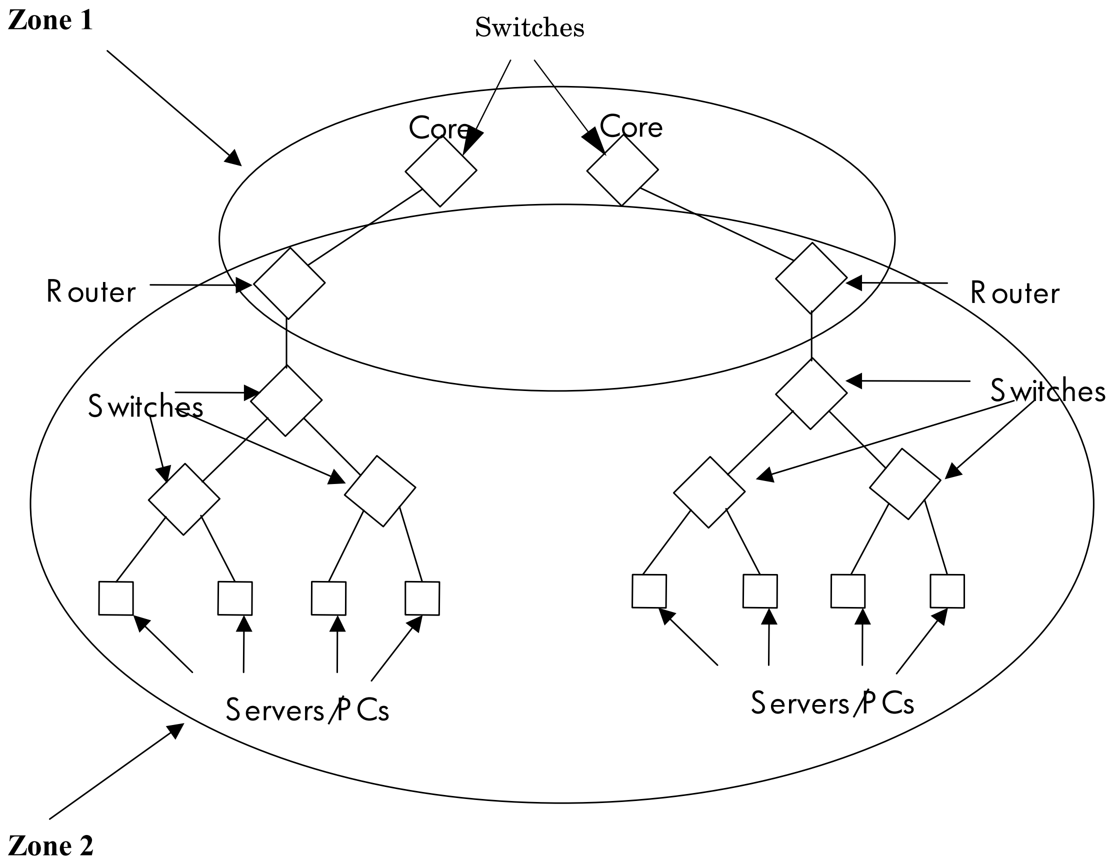
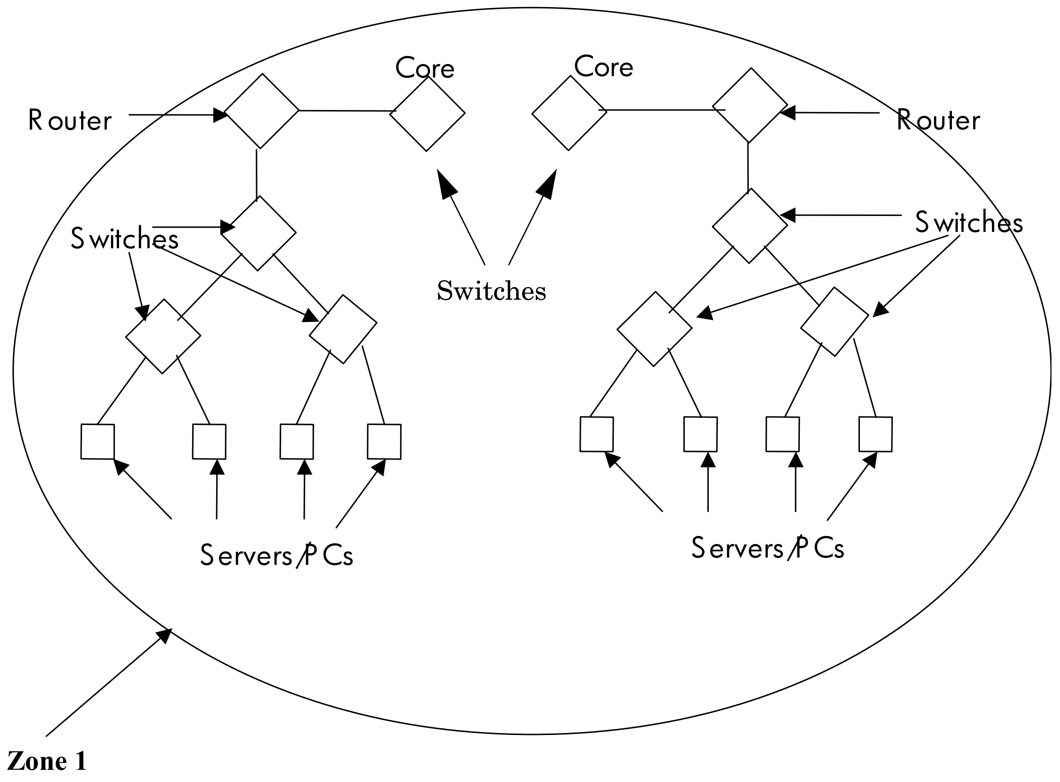


Figure 5 Core is switched: Using One Zone



Suggested Zone Configuration

- Refer to Figure 4 and Figure 5 for this discussion. To configure discovery for switch-to-switch, and switch-to-router connectivity, do the following:

  Follow the same specific guidelines as defined in Possibility 1 for getting switch-to-switch, and switch-to-router connectivity in the switch blocks of various buildings.

  Also, since the core devices are switches now, you should place all of the core devices and the corresponding distribution and WAN routers in the same zone for that building.

If your entire network is very small you can place all of your devices in one zone as shown in Figure 5. However, if Extended Topology recommended that you use multiple zones when you ran `setupExtTopo.ovpl`,use multiple zones.

If Zone 2 in Figure 4 exceeds a reasonable zone size, break up the zone into multiple zones.

- To configure discovery for router-to-router connectivity, do the following:

  Since the network type is mostly switched except for the distribution and WAN layers, and the distribution layer routers are connected to switches, the only router connectivity that is left to be discovered is WAN to WAN router connectivity. WAN routers typically will not run CDP, hence you can break up the WAN routers as you wish.

Automatic Zone Configuration

When you run the setupExtTopo.ovpl script, Extended Topology determines if there is a need for discovery zones. Extended Topology has the ability to automatically configure zones for larger environments. If you choose to have Extended Topology configure zones for you, make sure you follow the provided instructions carefully.

Use the following procedure to have Extended Topology automatically configure your zones. This procedure also tells you how to test your zones prior to running your first discovery:

1 Open the Extended Topology Configuration web page. See Open the Extended Topology Configuration Page on page 43 for more information.

2 Select the **Discovery Zones** tab.

3 If you did not already ask Extended Topology to automatically configure your zones in the setupExtTopo.ovpl script, you can request automatic configuration when you select **Configure Zones Automatically**.

  If you already ran the `setupExtTopo.ovpl` script and asked Extended Topology to automatically configure zones for you, do not reconfigure your zones at this time.

4 Select **Test All Zones** to test all of the zones, display any warnings, and view any suggested remedies. If necessary, manually reconfigure new zones to resolve the warning messages.

5 Be sure to select **Apply** to activate any manual changes.

After the zones are successfully configured, restart your Extended Topology discovery. See Start Extended Topology Discovery on page 23 for more information.

Verify and Modify Your Zone Configuration

After you configure your zones, if NNM discovers new nodes or if you manage nodes that were previously unmanaged, you may need to adjust your new zones. Extended Topology could place these nodes in the default zone or in an existing zone if the IP address matches one of the zone's subnet wildcards. Use the following techniques to decide if Extended Topology placed nodes into incorrect zones:

- Check the output for nodes that incorrectly appear in the default zone by using the following procedure:
  a. Open the Extended Topology Configuration web page. See Open the Extended Topology Configuration Page on page 43 for more information.
  b. Select the **Discovery Zones** tab.
  c. Select **Test All Zones** and review the displayed information.
- Check your topology for nodes that are missing connections.

After you identify misplaced nodes, you can manually place these nodes into the right zones or run another automatic zone configuration. See Manually Configure Zones on page 32 for more information.

Manually Configure Zones

In most cases, Extended Topology configures zones for you. However, in certain circumstances you may want to manually adjust your zone configuration. For example, you may need to adjust zones to eliminate warning messages or in an attempt to reduce Extended Topology's discovery time.

A good strategy for defining zones is to categorize your network devices by geography, such as by city, state, or building.

When defining your zones, do not separate switches that are directly connected together within a subnet.

To manually organize your devices into zones, use the following procedure:

1 Open the Extended Topology Configuration web page. See Open the Extended Topology Configuration Page on page 43 for more information.

2 Select the **Discovery Zones** tab.

3 Organize your devices into zones. You may need to calibrate your zones as outlined in this procedure. See Common Zone Configuration Examples on page 23 for more information.

   - Organize zones with fewer nodes when the nodes contain many interfaces. An example is a network that contains a high quantity of switches with many ports.

   - Organize zones with more nodes when the nodes contain fewer interfaces. An example is a network that contains a low quantity of switches with few ports.

4 You can limit Extended Topology discovery to only those devices you specify in zones. To do this, select the check box to exclude nodes that are not included in any of the zones you configure.

   ➤ When the Extended Topology discovery process begins, NNM transfers its node information to Extended Topology. Extended Topology includes these nodes in its discovery process. If you assign nodes to discovery zones, there may be a subset of the nodes transferred from NNM that are not included in any zone. Extended Topology automatically assigns these remaining nodes to a default zone. Select the check box to stop Extended Topology from discovering these nodes.

   You can also limit your discovery with the `bridge.noDiscover` file. Devices specified there are not discovered regardless of how you configure your zones. See Limit Extended Topology Discovery on page 44 for more information.

5 In the **Zone : Administration** text box, you can either specify any hostname that your DNS server can resolve to an IP address or directly specify any node's IP address. Separate entries with a semicolon. You can also use the following wildcard symbols:

- Asterisk: Use an asterisk to represent any number of characters up to the next period:

  *.corp.com matches pc.corp.com or ws.corp.com.

  pc.*.com matches pc.corp.com or pc.location.com.

  . matches corp.com or pc.com, but not pc.corp.com.

  The * in 10.*.1.3 matches any number.

- Question mark: Use to match a single character:

  pc.c?.com matches pc.ca.com, pc.cb.com, or pc.cc.com, but does not match pc.cal.com or pc.c.com.

  pc.???.com matches pc.abc.com, pc.bcd.com, or pc.cde.com, but does not match pc.ab.com or pc.abcd.com.

- Brackets: Use to match single characters, characters within a range, or characters not within a range:

  [bcf]an.corp.com matches ban.corp.com, can.corp.com, or fan.corp.com, but does not match dan.corp.com or lan.corp.com.

  [b-d]an.corp.com matches ban.corp.com, can.corp.com, or dan.corp.com, but does not match fan.corp.com, an.corp.com, or clan.corp.com.

  [!d-z]an.corp.com restricts the selection to aan.corp.com, ban.corp.com, or can.corp.com.

- Dash: Specify a range of IP addresses.

  10.2.1-3.1 represents 10.2.1.1, 10.2.2.1, or 10.2.3.1

6 Select **Add New Zone** to move each new zone into the **Current Zones** area of the **Extended Topology Configuration** screen. Select a zone, then select **Add to Zone** to add more addresses to a specific zone. You can also use **Delete** and **Replace** to help you manage your zones.

7 Select **Test All Zones** to test your zones. This test will check your zone configuration and may recommend configuration changes.

8 Select **Apply** to save your changes.

9 After the zones are successfully configured, restart Extended Topology discovery. See Start Extended Topology Discovery on page 23 for more information.

   After you move devices among zones, you must initiate a full Extended Topology discovery. If you add *new* devices to or delete devices from a single zone, you can save time by initiating an Extended Topology discovery on that single zone. See Discover a Single Zone on page 35 for more information.

10 You can check the amount of time Extended Topology spends discovering your zones by running the ovstatus -v ovet_disco command. If the discovery time of any of your zones is abnormally long when compared to that of other zones, do one or both of the following:

   - Add a new zone and move some of the devices from the problem zone into the new zone.
   - Move some of the devices from the problem zone into one or more of the existing zones that are being discovered faster.

   After your zones are successfully configured, remember to restart Extended Topology.

After Extended Topology completes a discovery with the new zone configuration, you should check the discovery results to make sure your zones are configured correctly. See Verify and Modify Your Zone Configuration on page 32 for more information.

Discover a Single Zone

If Extended Topology has completed a full discovery, you can rediscover an individual zone without initiating another full discovery. To discover devices in a single zone, complete the following steps:

1 Open the Extended Topology Configuration page. See Open the Extended Topology Configuration Page on page 43 for more information.

2 Select the **Discovery Zones** tab.

   If you configured Overlapping Address Domains, you can also select the **Overlapping Address Domain** tab and select a zone from that view.

3 Select the zone you want to discover.

4 Select **Discover Zone** to initiate a discovery of the selected zone.

The **Discover Zones** option causes Extended Topology to run a new discovery only for the devices in the specified zone. However, if you know of changes in multiple zones, you should run a full discovery.

After you initiate a discovery, you can monitor the status of your discovery. See Discovery Status on page 38 for more information.

How Extended Topology Uses Classic NNM Information

Extended Topology uses NNM data to speed up discovery. When a new discovery begins, Extended Topology starts background processes that retrieve data from NNM about the devices NNM is actively managing. Extended Topology discovers information from these devices if they respond to SNMP requests.

For a device that does not respond to SNMP requests, Extended Topology creates a node and a local interface for that node using only NNM data. This node appears in the Extended Topology views, but it may contain an incomplete set of attributes and connectivity. To identify an unresponsive node, move your mouse over the node in one of the Extended Topology views. Extended Topology will display a message about SNMP access problems with the node.

The following information is copied from NNM to Extended Topology and used by several Extended Topology processes. See Figure 6 on page 37 for a graphical illustration.

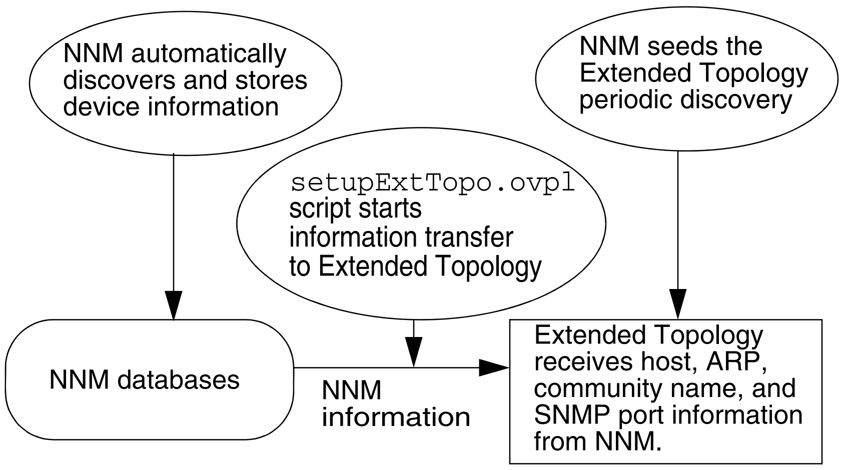
- NNM IP address and hostname information.
- NNM ARP cache information.
- NNM SNMP community string information. See the *ovsnmp.conf_db* reference page for more information. Details about how to access reference pages are in the online help.
- NNM SNMP port information.

Extended Topology uses the information it receives from NNM to calculate the number of nodes it may discover. Extended Topology notifies you when the number of discovered nodes exceeds the number of nodes it is licensed for. Extended Topology may discover additional interfaces after receiving information from NNM, however it does not count these interfaces as discovered nodes for licensing purposes.

Use the **Options** → **Configuration** → **SNMP Configuration** menu to change device community string information in NNM. Extended Topology applies SNMP community string configuration changes during the next discovery cycle. See the *xnmsnmpconf* reference page for more information. Details about how to access reference pages are in the online help.

To apply the new SNMP community string changes immediately, run a new Extended Topology discovery. See Start Extended Topology Discovery on page 23 for more information.

Figure 6 NNM to Extended Topology Information Transfer


NNM automatically discovers and stores device information
NNM seeds the Extended Topology periodic discovery
setupExtTopo.ovpl script starts information transfer to Extended Topology
NNM databases
NNM information
Extended Topology receives host, ARP, community name, and SNMP port information from NNM.

Discovery Status

You must wait for Extended Topology to complete an initial discovery before using Extended Topology views. To view Extended Topology discovery status, use the following procedure:

1 Open Home Base. See Launch Home Base on page 86 for more information.

2 Select the **Discovery Status** tab.

Discovery status is normally updated frequently, but there can be long periods, potentially many hours, where the progress bar does not change. This is due to the duration of some processing steps and the way progress is measured. It does not reflect a problem with discovery.

While Extended Topology discovery is in progress, the activity indicator moves to indicate that everything is proceeding normally. The activity indicator usually moves every few seconds, but you should keep in mind that it may pause during lengthy internal operations. Ordinarily, a pause of this kind will be fifteen minutes or less.

However, if the activity indicator is stationary for too long, such as an hour or more, Extended Topology discovery may have stalled. You can use the discovery status time stamp in the text area to find the time of the most recent update.

You can use the following command to display the status of each Extended Topology discovery process.

- *UNIX*:

```
$OV_BIN/ovstatus -c
```

- *Windows*:

```
%OV_BIN%\ovstatus -c
```

Many Extended Topology processes only run during discovery. After Extended Topology completes its discovery, these processes automatically exit. These processes show the following process status output when you run the ovstatus -c command:

```
ovet_processname - NOT_RUNNING    Exited and awaiting next
discovery. Exit (0).
```

Extended Topology automatically restarts its discovery processes and begins another discovery when it reaches or exceeds its discovery thresholds. You can use the Extended Topology Configuration web page to modify these discovery thresholds. See Open the Extended Topology Configuration Page on page 43 for more information.

Recurring Discovery

The Extended Topology model of recurring discovery differs from the continuous discovery present in NNM. Extended Topology captures data about your network as it exists during the most recent discovery cycles. It does not update the data between discovery cycles.

This means that only the layer 2 connections that exist during Extended Topology discoveries are recorded. In addition, device information is gathered only from devices that are accessible during the Extended Topology discoveries. An accessible device responds to SNMP requests from NNM. For this to happen, NNM must be configured to use the correct SNMP GET-Community name for each device you want to discover.

Situations may arise that prevent a previously discovered device from responding with device information during later discovery cycles. When a situation like this arises, Extended Topology retains data about the previously discovered but unresponsive device from recent discovery cycles. This data is retained for 8 days.

In contrast, if a specific device becomes unresponsive and continues to be unresponsive during several consecutive discovery cycles, the device and its information are no longer shown in Dynamic Views.

Post-discovery changes in the network are not dynamically reflected in the data that Extended Topology provides. Rather, the changes are incorporated during the next discovery cycle. Implications of this behavior are as follows:

- If a device was inaccessible during recent Extended Topology discoveries but responded with device information during the most recent discovery cycle, a VLAN view will display VLAN information for it.
- If a device became inaccessible during some past Extended Topology discovery and continued to be unresponsive during several consecutive discovery cycles, a VLAN view will not display the device or VLAN information about the device.

- A Neighbor view will omit a neighboring device that was inaccessible during several consecutive discovery cycles.

This list is not comprehensive, but gives you a sense of how periodic discovery can affect the data you observe later.

You can modify Extended Topology's default discovery options to meet your specific needs using information from this list. See Extended Topology Discovery Process on page 41 for more information.

Configure Extended Topology Discovery

In smaller environments, Extended Topology begins discovering network information after you run the setupExtTopo.ovpl script. Extended Topology starts a discovery after a default number of NNM topology changes occur.

Extended Topology Discovery Process

To modify Extended Topology discovery options, use the Extended Topology Configuration web page. See Open the Extended Topology Configuration Page on page 43. You have several configuration options. You can configure Extended Topology discovery behavior if you select the **Discovery Behavior** tab. This allows you to do the following:

- Initiate a new discovery every time Extended Topology is restarted.
- Enable or disable Extended Topology recurring discovery.
- Begin a new discovery after NNM detects a number of topology changes. The number of topology changes includes layer 3 discovery information from NNM.
- Schedule a new discovery daily or weekly.
- Immediately begin a new discovery by selecting **Initiate Full Discovery Now**.

Make sure you select Apply to save your changes.

➤ If you configure Extended Topology to initiate a new discovery every time Extended Topology is restarted, every time you run the following command, a new Extended Topology discovery is initiated.

- *UNIX*:

```
$OV_BIN/ovstart
```

- *Windows*:

```
%OV_BIN%\ovstart
```

Use caution when using the *ovstart* command because it restarts all NNM and Extended Topology processes. See the *ovstart* reference page for more information. Details about how to access reference pages are in the online help.

After you modify the Extended Topology discovery options, take one of the following actions to apply your changes:

- If you made changes that modify multiple zones, start a new Extended Topology discovery. See Start Extended Topology Discovery on page 23 for more information.
- If you added new devices to a specific zone, you only need to discover that zone. See Discover a Single Zone on page 35 for more information.

You can display the process statuses by executing the following command:

- *UNIX*:

```
$OV_BIN/ovstatus -c
```

- *Windows*:

```
%OV_BIN%\ovstatus -c
```

If you restart Extended Topology processes and you do not have Extended Topology configured to run a new discovery every time you restart it, the ovstatus -c command output is similar to the following:

```
ovet_disco 5621       RUNNING      Discovery Completed: <date and time
                                   of last discovery>.

ovet_bridge 22321     RUNNING      Extended Topology process management
                                   occurred.

ovet_dhelpserv 22464  RUNNING      Initialization complete.

ovet_processname -    NOT_RUNNING  Exited and awaiting next discovery.
                                   Exit(0)
```

The RUNNING portion of the message indicates a state of readiness for discovery processes. A new discovery does not occur until one of the following happens:

- Extended Topology meets the discovery configuration parameters you set on the Extended Topology Configuration page.
- You manually restart discovery. See Start Extended Topology Discovery on page 23 for more information.

Open the Extended Topology Configuration Page

To open the Extended Topology Configuration web page, use one of the following methods:

- From NNM, select the **Options** → **Extended Topology Configuration** menu.
- From Home Base, select the **Discovery Status** tab and then select the **Extended Topology Configuration** button.

Limit Extended Topology Discovery

You can exclude devices from Extended Topology discovery by creating an Extended Topology Discovery Exclusion List.

Extended Topology Discovery Exclusion List

Extended Topology limits the breadth of discovery according to the contents of the following file:

- *UNIX*:

  ```
  $OV_CONF/nnmet/bridge.noDiscover
  ```

- *Windows*:

  ```
  %OV_CONF%\nnmet\bridge.noDiscover
  ```

To exclude devices from discovery, enter NNM filter names, IP addresses, and wildcards in the bridge.noDiscover file.

Here are a few examples of valid bridge.noDiscover file entries:

- **10.2.112.86** # Exclude this IP address from discovery.
- *.*.*.* # Exclude all nodes from discovery.
- **10.2.*.*** # Exclude all IP addresses from 10.2.0.0 to 10.2.255.255.
- **10.2.0-2.*** # Excludes all nodes from 10.2.0.0 to 10.2.2.255.
- **Routers** #Excludes all nodes matching the NNM filter *Routers*.

Do the following to create the bridge.noDiscover file:

1. As Administrator or root, create the bridge.noDiscover file.
2. Add NNM filter names, IP addresses, or wildcards you want excluded by Extended Topology discovery. Enter one NNM filter name, IP address, or wildcard per line.
3. Run a new Extended Topology discovery.

See the *bridge.noDiscover* reference page for more information. Details about how to access reference pages are in the online help.

Incremental Discovery

You can configure Extended Topology discovery incrementally discover nodes and interface configuration changes. This section covers these topics.

⚠ Incremental discovery is not a substitute for full or zone discovery. After several configuration changes, you need to run a full or zone discovery.

Before you can use the incremental discovery feature, you must complete at least one successful discovery.

Node Discovery

Single nodes that are added to classic NNM topology are automatically made a part of Extended Topology without a full or zone rediscovery. At this time, only new nodes and their interfaces are added to Extended Topology. Connectivity for newly added nodes is not discovered.

The nodes, interfaces, and addresses that are added to Extended Topology are monitored by APA. Since the interfaces are unconnected, polling is controlled by a customer filter that uses the new isNewNode and isNewInterface attributes on the newly added nodes. After a full or incremental zone discovery occurs for these nodes, the isNewNode and isNewInterface attributes are cleared in the ET topology, and normal APA filtering configuration applies.

Turn On Incremental Node Discovery

By default, incremental node discovery is turned off. To turn it on, complete the following steps:

1. Run the following command to stop the bridge process:

   ```
   ovstop -c ovet_bridge
   ```

2. Open the following file for editing:

   - *UNIX*:

     ```
     $OV_LRF/ovet_bridge.lrf
     ```

   - *Windows*:

     ```
     %OV_LRF%\ovet_bridge.lrf
     ```

3 Find the following line in the `ovet_bridge.lrf` file:

```
OVs_YES_START:ovtopmd::OVs_WELL_BEHAVED:15:NOPAUSE
```

4 Add the bolded text to the line as follows:

```
OVs_YES_START:ovtopmd:-single_node_discovery:OVs_WELL_BEH
AVED:15:NOPAUSE
```

5 To activate your changes, run the following command:

```
$OV_BIN/ovaddobj ovet_bridge.lrf
```

6 Run the following command to start the bridge process:

```
ovstart -c ovet_bridge
```

Process New Nodes

The Extended Topology process, `ovet_bridge`, must be running in order for new nodes to be added to topology. If `ovet_bridge` is not running when new nodes are added, these nodes are processed after `ovet_bridge` is started again.

`ovet_bridge` verifies the existence of NNM Standard Edition nodes when configuration polls are performed. You can manually trigger these activities with the Demand Poll operation in either of the following ways:

- Select Demand Poll from the NNM user interface Fault menu.
- Use the command line tool, nmdemandpoll.

Logs

Key incremental node discovery activities are logged in the following file:

```
$OV_PRIV_LOG/ovet_bridge.log
```

Known Problems

Duplicate and secondary interfaces from the NNM/SE topology are not reflected in NNM/AE Topology.

- Duplicate Interfaces. When netmon encounters switches, it creates duplicate interface objects to aid in attaching the device to its various segments. These interfaces are tagged with "NETMON_MADE" in the NNM/SE topology and are visible when you run the following command at a command prompt: `ovtopodump -l <node-or-interface-id>`
- Secondary Interfaces. In MIB-II, multiple entries in ipAddrTable can refer to the same ifTable entry. This means that an interface has multiple IP addresses. Due to limitations of the NNM/SE topology model, an interface object is created for each of these ipAddrTable references. ovet_bridge only creates an interface for one of these duplicates. The interface uniqueness is determined by the interface name: `<node-name>[ 0 <ifIndex> ]`

Interface Configuration Change Discovery

Extended Topology discovery periodically discovers your network. The following network configuration changes that occur between discovery periods can be dynamically updated if you make the configuration changes described in the Interface Configuration Changes and Interface Renumbering sections.

If you choose to enable this feature, you should run a complete discovery after you enable the feature. This action synchronizes discovery and the network before changes are detected and updated.

- Interface Configuration Changes
  - Alias updates
  - IP address assignments
  - Board/interface addition and removal
- Interface Renumbering
  - Interface index updates

The following must be true to ensure success for this functionality:

- APA must be enabled as the default poller.
- Netmon must be running with the `-k useIfAlias=true` option.
- The interface ifDescr (1.3.6.1.2.1.2.2.1.2) value must be unique and not null for every interface on a device.

Interface Configuration Changes

By default, the automatic discovery of interface configuration changes is disabled.

To enable the automatic discovery of interface configuration changes, complete the following steps:

1 Run the following command to stop the `ovet_bridge` process:

```
ovstop -c ovet_bridge
```

2 Open the following file for editing:

- *UNIX*:

  ```
  $OV_LRF/ovet_bridge.lrf
  ```

- *Windows*:

  ```
  %OV_LRF%\ovet_bridge.lrf
  ```

3 Find the following line in the `ovet_bridge.lrf` file:

```
OVs_YES_START:ovtopmd::OVs_WELL_BEHAVED:15:NOPAUSE
```

4 Add the bolded text to the line as follows:

```
OVs_YES_START:ovtopmd:-interface_config_change:OVs_WELL_B
EHAVED:15:NOPAUSE
```

5 To activate your changes, run the following command:

```
$OV_BIN/ovaddobj ovet_bridge.lrf
```

6 Run the following command to start the `ovet_bridge` process:

```
ovstart -c ovet_bridge
```

Interface Renumbering

Some device configuration changes require that you reboot a device to activate the changes. This can result in interface renumbering. This remapping can be automatically detected.

By default, the automatic discovery of interface renumbering changes is disabled.

To enable the automatic discovery of interface renumbering, complete the following steps:

1 Run the following command to stop the `ovet_bridge` process:

```
ovstop -c ovet_bridge
```

2 Open the following file for editing:

- *UNIX*:

  ```
  $OV_LRF/ovet_bridge.lrf
  ```

- *Windows*:

  ```
  %OV_LRF%\ovet_bridge.lrf
  ```

3 Find the following line in the `ovet_bridge.lrf` file:

```
OVs_YES_START:ovtopmd::OVs_WELL_BEHAVED:15:NOPAUSE
```

4 Add the bolded text to the line as follows:

```
OVs_YES_START:ovtopmd:-handle_interface_renumber::OVs_WEL
L_BEHAVED:15:NOPAUSE
```

5 To activate your changes, run the following command:

```
$OV_BIN/ovaddobj ovet_bridge.lrf
```

6 Run the following command to start the `ovet_bridge` process:

```
ovstart -c ovet_bridge
```

Interface Configuration Change Detection Timing

Interface configuration change discovery uses APA alarms as a trigger. When specific APA events occur, attributes are automatically updated. See Interface Configuration Change Detection on page 149 for more information. Both the classic NNM and Extended Topology databases are updated.

As explained in the list below, the time it takes to discover an interface configuration change depends on the time it takes for APA or netmon to detect the change.

- Alias updates

  Alias updates are detected when APA queries ifAlias attributes in the configuration polling cycle.

- IP address assignments

  Changes in IP address assignments are detected when netmon does a configuration poll. You can trigger a netmon configuration poll with the following command: `nmdemandpoll`.

- Board/interface addition and removal

  Board/interfaces that are added or removed are detected by APA in the status polling cycle.

- Interface index updates

  APA monitors the following MIB OIDs:

  — `sysUpTime` (1.3.6.1.2.1.1.3) - enabled by default

  — `entLastChangeTime` (1.3.6.1.2.1.47.1.4.1) - disabled by default

  For information on enabling the `entLastChangeTime` MIB OID, see Enable Interface Configuration Change Detection on page 150.

  If either of these values change, the change triggers an investigation that can result in an interface change alarm.

For information on adjusting APA polling settings, see the *Extended Topology and APA Deployment Guide*. For information on adjusting netmon's polling settings, see the *Managing Your Networks* manual.

Limitations

- Discovered protocol data such as VLANs, meshes, xRP, and MPLS are not updated with the interface configuration changes.
- Added interfaces exist as unconnected interfaces until the next discovery cycle. These interfaces are polled according to the polling polices applicable for isNewInterface.
- Interface filtering is not honored.
- Interface configuration change discovery does not work if the IPT SPI is installed.

Connectivity

This section discusses options you have to improve the connectivity Extended Topology displays.

Connect End Nodes from Another Zone

If you do a full Extended Topology discovery, connections for end nodes can be discovered when a switch or router and its corresponding end nodes are in different zones.

➤ Any node that is not a router or a switch is considered an end node.

To configure discovery to derive connections for end nodes in a different zone, complete the following steps:

1 Open the `DiscoSchema.cfg` file for editing. You can find this file at the following location:

- *UNIX*:

  ```
  $OV_CONF/nnmet/DiscoSchema.cfg
  ```

- *Windows*:

  ```
  %OV_CONF%\nnmet\DiscoSchema.cfg
  ```

2 Find the block that defines the disco.config table inserts, and replace the value for `m_UseInferenceLogic` to `65`. Following is an example. The changed value is in bold.

```
insert into disco.config
(
   m_NothngFndPeriod,
   m_AvgeFindPeriod,
   m_RestartAgents,
   m_RestartFinders,
   m_DirScanIntvl,
   m_AgentWaitPeriod,
   m_UseInferenceLogic,
   m_DiscoverDefaultVlans,
   m_DiscoverInternalVlans
)
values
(
   10,
   0.25,
   0,
   0,
   600,
   3600,           // seconds
   65,
   0,
   0
);
```

The values for `m_UseInferenceLogic` are additive and are used as bitmask values in the connectivity algorithm. The default setting is 1.

To use this feature exclusively without the default setting, you can specify a value of 64; however, a value of 65 is recommended so you do not lose the functionality provided by the default setting. This logic is illustrated as follows:

64 = end node connectivity across zones

1 = default settings

To gain end node connectivity across zones and retain the default setting functionality, add the values and set the value to 65.

3. Save and close the modified file.
4. Start a full Extended Topology discovery to activate your changes. You can initiate a full discovery on the Extended Topology Configuration web page. See Open the Extended Topology Configuration Page on page 43 for more information.

End Node Connection Troubleshooting

If you have modified the `DiscoSchema.cfg` file and end node connections still do not exist, use the following steps to troubleshoot the problem:

1. Is the switch, to which the end node is physically connected, seeded into Extended Topology discovery? Look for the IP address of the switch in the `hosts.nnm` file. You can find this file at the following location:

   UNIX:

   ```
   $OV_DB/nnmet/hosts.nnm
   ```

   Windows:

   ```
   %OV_DB%\nnmet\hosts.nnm
   ```

   - If the switch is in the `hosts.nnm` file but the end node connection is missing, complete the following actions:
     - Verify that the switch responds to SNMP with a `snmpwalk` command.
     - If the switch responds to SNMP, verify that the switch reports the end node by checking the end node IP in the following file:

       UNIX:

       ```
       $OV_DB/nnmet/ReturnsDump.txt.nodup
       ```

       Windows:

       ```
       %OV_DB%\nnmet\ReturnsDump.txt.nodup
       ```

       In the sample record from `ReturnsDump.txt` below, **`10.150.156.1`** is the Switch IP, and **`10.155.156.10`** is the End Node IP:

       ```
       10.150.156.1,1,00:10:7B:80:EF:44,0,10.155.156.10,00
       :10:29:E6:13:D8
       ```

    – If the switch responds to SNMP and reports the end node, verify that the end node name is in the following file:

      UNIX:

      ```
      $OV_DB/nnmet/MacIndex.txt.nodup
      ```

      Windows:

      ```
      %OV_DB%\nnmet\MacIndex.txt.nodup
      ```

      If the end node name is not present, it means that the end node is not responding with interface details so an interface cannot be created for this node. If there is no interface data, the connection cannot be created in the database.

- If the switch is not in the `hosts.nnm` file, complete the following actions:

  — Is the switch in the NNM discovery database? Use the `ovtopodump` command. See the *ovtopodump* reference page for more information. Details about how to access reference pages are in the online help.

  If the switch is not in the NNM discovery database, go to step 2.

2 If the switch is not in the NNM discovery database, use the `loadhosts` command to load the switch, and then complete an `nnmdemandpoll` to speed discovery. See the *loadhosts* and *nmdemandpoll* reference pages for more information. Details about how to access reference pages are in the online help.

If the switch is in the NNM discovery but is not seeded into *hosts.nnm*, make sure the switch is not intentionally excluded from discovery. See Limit Extended Topology Discovery on page 44 for more information.

Layer 3 Edge Connectivity

When you launch Dynamic Views, connections may be present that do not actually exist. These erroneous connections come from the Extended Topology database.

A source for erroneous connections are layer 3 connections that are derived from IP subnet analysis. At the end of each Extended Topology discovery cycle, the EdgeL3Conn utility is called. This utility reviews all the subnet entries in the Extended Topology database and looks for a subnet that is shared by two

IP addresses/interfaces. When the utility finds a subnet that is shared by only two interfaces, it assumes that the two interfaces are the interfaces of two edge routers and adds a connection between them.

Turn Off Layer 3 Edge Connectivity

To turn off the behavior that causes these erroneous connections, complete the following steps:

1. Make a backup copy of the following file:
   - *UNIX*:

     ```
     /etc/opt/OV/share/conf/nnmet/EdgeL3Conn.cfg
     ```

   - *Windows*:

     ```
     <install_dir>\conf\nnmet\EdgeL3Conn.cfg
     ```

2. Open the `EdgeL3Conn.cfg` file for editing, and find the following line:

   ```
   #enableConnectivity:1
   ```

3. To prevent the erroneous connections, remove the comment indicator and change the value for enableConnectivity as follows:

   ```
   enableConnectivity:0
   ```

4. To activate your changes, run a new Extended Topology discovery. See Start Extended Topology Discovery on page 23 for more information.

Evaluate Erroneous Connections

To determine if an erroneous connection is the result of IP subnet analysis, complete the following steps:

1. Run the following command:
   - *UNIX*:

     ```
     /opt/OV/support/NM/ovet_topoquery getAllSubnets >
     getAllSubnets.out
     ```

   - *Windows*:

     ```
     <installdir>/support/NM/ovet_topoquery getAllSubnets >
     getAllSubnets.out
     ```

2 Examine the output file, getAllSubnets.out.

For each subnet that has a PrefixLength that is equal to or greater than the value specified for the minAllowedBitmask parameter in the `EdgeL3conn.cfg` file, look at the list of the subnet end points under the SubnetEndPoint section. If exactly two end points are specified, a connection is created between the IP interfaces.

Connection Editor

In certain circumstances where device information is limited, Extended Topology may not accurately discover and model every connection in a network. As a result, you may see no connections where you know connections exist or connections indicated where you know none exist.

Be careful in your assessment of connection accuracy. Extended Topology frequently surprises network administrators by showing that the network's actual topology differs from expectations. Before using the facilities described here, make sure that your changes reflect the true topology, and not merely what is believed about it.

To modify Extended Topology's connection information, you first create the connectionEdits file, and then add and delete specific connections by making entries in that file. This chapter describes the procedure in detail.

Some device configuration activities cause SNMP agents to renumber the interfaces of some Ethernet switches and routers. These activities include the following actions:

- Moving a board from one slot to another
- Removing a board
- Adding a new board
- Removing the supervisor board in a dual supervisor board system
- Power cycling a switch or router

You should not use the `connectionEdits` file to modify the discovered connections for these nodes. If you do, you will have to modify the entries for that device in the `connectionEdits` file each time the device's interfaces are renumbered.

At the end of its work, the ovet_disco process uses the connectionEdits file to modify the discovered topology data. You can also apply connection edits to the data from a previous discovery using the ovet_topoconnedit.ovpl script.

Use the Connection Editor

Extended Topology uses the ovet_disco process to collect connectivity information from network infrastructure devices (connectors) and other non-connector devices (end nodes) such as servers, workstations, or PCs. This information includes device attributes and their relationships to other devices in the network.

Extended Topology uses this information to show you how your devices interconnect. You can amend or remove device connections during the course of a discovery, or after a discovery has completed by creating and modifying the following file:

- *UNIX*:

  ```
  $OV_DB/nnmet/connectionEdits
  ```

- *Windows*:

  ```
  %OV_DB%\nnmet\connectionEdits
  ```

connectionEdits File Syntax

You can describe connections in the connectionEdits file using the names of interfaces located on connectors and end nodes. These interfaces must already exist in the Extended Topology, or must be discovered by the ovet_disco process.

You can add interfaces to the connectionEdits file using the following form:

```
fully-qualified-node-name[ 0[ ifIndex ] ]
```

An example of this is show below:

```
coreswitch3.corp.net[ 0[ 50 ] ]
```

Use OQL Inserts

Interface specifications are combined into SQL statements, or *inserts*. More specifically, these inserts are OQL statements, which are a subset of SQL statements. These OQL statements have extensions for controlling Extended Topology operations and are used internally during Extended Topology discovery to create or delete connection representations. The following example shows the format of an OQL insert statement in the connectionEdits file:

```
insert into disco.connectionEdits (m_Name, m_NbrName,
m_Command) values ('interface1-name', 'interface2-name',
command);
```

Below are some hints on how to add inserts to the connectionEdits file:

- The quotes and semicolon are required.
- Each insert statement must be on a single line.
- The order of the interfaces in the OQL statement is not important. Extended Topology checks the connectivity information from both devices referenced in the insert statement.
- You can use one of the following entries as a *command* parameter:
  - 0 - This tells Extended Topology to ignore this entry. This is helpful if you want to place comments in your entries.
  - 1 - This tells Extended Topology to add this connection.
  - 2 - This tells Extended Topology to delete this connection.

OQL Insert Examples

Below are some examples of how you might use the connectionEdits file to meet your specific needs. Each example uses connected devices, Switch A and Switch B, with ifIndex 91.

- If you want to comment on the connection between the two connected devices, but you do not want Extended Topology to take any action, add the following line to the connectionEdits file. Notice that the expectation for no action is designated by “0” at the end of the line.

```
insert into disco.connectionEdits (m_Name, m_NbrName,
m_Command) values ('SwitchA.hp.com[ 0 [ 91 ] ]',
'SwitchB.hp.com[ 0 [ 91 ] ]', 0);
```

- If you want to configure Extended Topology to show the connection between two connected devices, add the following line to the connectionEdits file. Notice that the only change from the first example is to change the “0” to “1” at the end of the line to indicate that a connection should be inserted.

  ```
  insert into disco.connectionEdits (m_Name, m_NbrName,
  m_Command) values ('SwitchA.hp.com[ 0 [ 91 ] ]',
  'SwitchB.hp.com[ 0 [ 91 ] ]', 1);
  ```

- If you want to remove an invalid connection between two devices, add the following line to the connectionEdits file. Notice that the only change from the first example is to change the “0” to “2” at the end of the line to indicate that a connection should be removed.

  ```
  insert into disco.connectionEdits (m_Name, m_NbrName,
  m_Command) values ('SwitchA.hp.com[ 0 [ 91 ] ]',
  'SwitchB.hp.com[ 0 [ 91 ] ]', 2);
  ```

Helpful Tools

There are several tools that can help you create and modify the connectionEdits file. These tools are located in the following directories:

- *UNIX*:

  ```
  $OV_MAIN_PATH/support/NM
  ```

- *Windows*:

  ```
  %OV_MAIN_PATH%\support\NM
  ```

The tools in the above directory are made available for HP support engineers, and are not designed or tested for end-users. They are described here because you may find them helpful, but Hewlett-Packard offers no support for them.

Identify the ifIndex

Switch vendors use various schemes to map physical ports to ifIndex values. Some vendors number their ports sequentially starting at 1 and have a one-to-one mapping of port numbers to ifIndex values. Some vendors label the port numbers based on its position on the board on which it is located. For example, *3/1* would represent *board 3, port 1*, and *B2* would represent *board B, port 2*. Then the vendors use a scheme to map the board/port combinations to ifIndex values.

Extended Topology discovers these board/port-to-ifIndex relationships. In many cases, switches also incorporate the board and port information into the ifDescr or ifName objects reported by the SNMP agent. To see these relationships, you can use the following command:

```
ovet_topoquery getNodeByName node-name
```

Look for the *Description* in the NMInterface sections of the output or the IfName, BoardNo and PortNum fields in the Interface Property sections of the output. Use the IfIndex or EntityName field of that output to determine the appropriate ifIndex value to use in the connectionEdits file.

Routers and end-nodes typically have IP addresses associated with their connected interfaces. NNM and Extended Topology use the value supplied in the SNMP MIB table ipaddrTable to link IP addresses to ifIndex. You can use the following command to see the IP-address-to-ifIndex association for discovered interfaces:

```
ovtopodump -l <node-name>
```

Use Support Tools to Assist You

There are several support tools you can use when creating and adding insert statements into the connectionEdits file. You must run these tools as Administrator or root.

- *ovet_topoconndump.ovpl script*

  You can use the ovet_topoconndump.ovpl script prior to adding insert statements to the connectionEdits file. When used without any parameters, it will dump all the layer 2 connections currently in the Extended Topology in the form of OQL insert statements into the disco.connectionEdits table.

  You can also use the -node parameter with the ovet_topoconndump.ovpl script to display the layer 2 connections associated with a specified node. The syntax would look like this:

  ```
  ovet_topoconndump.ovpl-node fully-qualified-node-name
  ```

  Below are some examples of how to use the ovet_topoconndump.ovpl script:

  — If you run the ovet_topoconndump.ovpl script with no arguments, Extended Topology displays all layer 2 connections in the Extended Topology.

— If you run ovet_topoconndump -node hp4k1sw.hp.com, Extended Topology displays the layer 2 connections associated with hp4k1sw.hp.com. You can capture the text from this display and use it as a starting point for creating your connectionEdits file. The output would look something like this:

```
insert into disco.connectionEdits (m_Name, m_NbrName,
m_Command) values ('hp4k1sw.hp.com[ 0 [ 91 ] ]',
'hp4k2sw.hp.com[ 0 [ 91 ] ]', 0);
insert into disco.connectionEdits (m_Name, m_NbrName,
m_Command) values ('hp4k1sw.hp.com[ 0 [ 81 ] ]',
'hp212sw.hp.com[ 0 [ 18 ] ]', 0);
insert into disco.connectionEdits (m_Name, m_NbrName,
m_Command) values ('cisco5500.hp.com[ 0 [ 6 ] ]',
'hp4k1sw.hp.com[ 0 [ 17 ] ]', 0);
```

➤ Notice that Extended Topology includes the number 0 (the first character to the right of the left square bracket following each fully qualified node name) in each entry line item. Do not remove or modify the number 0 when adding inserts into your connectionEdits file. The 0 character is reserved for future use and is not currently used.

- ovet_topoconnedit.ovpl script

  You can use the ovet_topoconnedit.ovpl script to make changes in an existing Extended Topology without running a discovery. The ovet_topoconnedit.ovpl script reads the connectionEdits file and updates the Extended Topology database. Extended Topology silently ignores invalid or duplicate entries contained within the connectionEdits file.

  You must stop and restart the embedded OpenView application server, ovas, to see the results of the applied changes. You should also restart the ovet_poll process in order to inform it of any interface connectivity changes from the connectionEdits file. See Stop and Restart Processes on page 17 for more information.

Practical Examples

The following examples depict some actual connectivity problems and potential solutions using the connectionEdits file.

Map Modifications

Refer to Figure 7 on page 63 for this example. Suppose Extended Topology "discovers" a connection between Switch A and Switch B that does not actually exist. For a number of reasons, including accurate root-cause analysis, you want to remove that connection from topology.

You can use the following procedure to remove the connection between Switch A to Switch B from future discoveries:

1 As Administrator or root, run the following command:

```
ovet_topoconndump.ovpl -node hp4k1sw.hp.com
```

Running the above command results in the following display:

```
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('SwitchA.hp.com[ 0 [ 1 ]
]','hp4k1sw.hp.com[ 0 [ 56 ] ]',0);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('SwitchB.hp.com[ 0 [ 91
] ]','hp4k1sw.hp.com[ 0 [ 91 ] ]',0);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('hp4k1sw.hp.com[ 0 [ 91
] ]','hp4k2sw.hp.com[ 0 [ 91 ] ]',0);
```

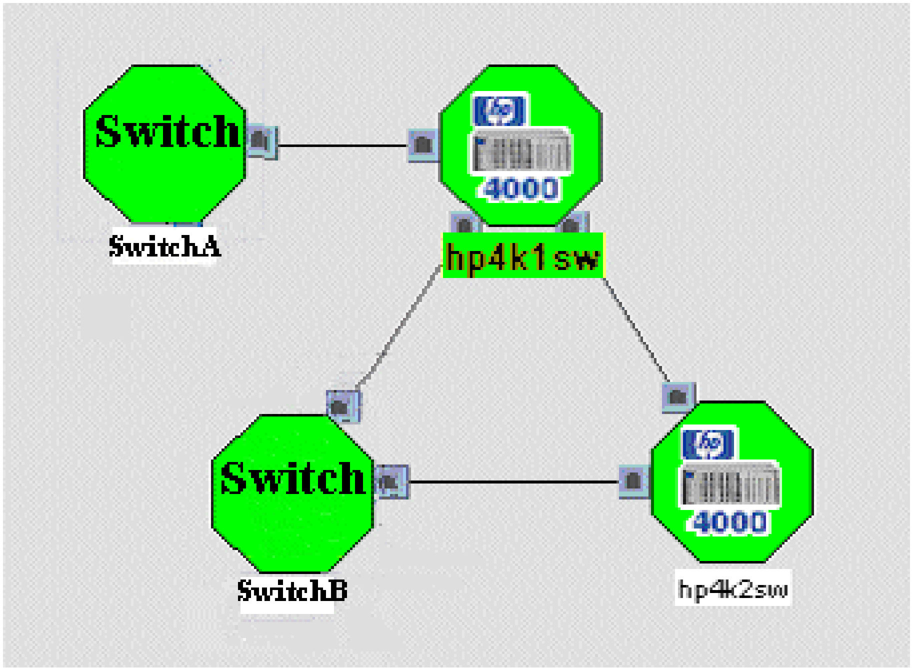
Notice that the above display that describes a connection between ifIndex 56 of hp4k1sw to ifIndex 1 of Switch A.

2 Edit the connectionEdits file and add the following text:

```
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('SwitchA.hp.com[ 0 [ 1 ]
]','hp4k1sw.hp.com[ 0 [ 56 ] ]',2);
```

Notice that the value for the m_Command field of the insert statement is 2 for delete the connection.

Figure 7 Making Map Modifications



Add Connections that are Missing

Suppose, in Figure 7, that Extended Topology failed to discover an actual connection between Switch B and Switch A.

You can use the following steps to add this connection to future discoveries:

1 Determine which ifIndex value of Switch B to use for the connection. For this example, assume that port B2 is used on Switch B. As Administrator or root, run the following command:

```
ovet_topoquery getNodeByName SwitchB.hp.com
```

Running the above command results in the following display:

```
+++++++++++++++++ NMInterface ++++++++++++++
ObjID:2d037786-f5c1-71d7-11b6-0f0276230000
NNMObjID:764
EntityName:SwitchB.hp.com[ 0 [ 42 ] ]
Description:B2
:
:
==========Interface Property===============
VPI:0
VCI:0
BoardNo:B
PortNum:2
AuxPortNum:42
IfIndex:42
IfName:B2
IfAlias:-
IfType:6
PhysicalAddress:00:30:C1:EF:13:D7
:
:
```

By looking at the information listed in the above output, you determine that you should use ifIndex value 42 for Switch B.

2 Determine which ifIndex value of Switch A to use for the connection. As Administrator or root, run the following command: ovet_topoquery getNodeByName SwitchA.hp.com

Running the above command results in the following display:

```
+++++++++++++++++ NMInterface ++++++++++++++
ObjID:31ebc672-f5c1-71d7-11b6-0f0276230000
NNMObjID:808
EntityName:SwitchA.hp.com[ 0 [ 2 ] ]
Description:2
:
:
==========Interface Property===============
VPI:0
VCI:0
BoardNo:
PortNum:2
AuxPortNum:2
IfIndex:2
IfName:2
:
:
```

3 For this example, assume that port 2 is the actual port number used on SwitchA as displayed in the output from the previous step. Edit the connectionEdits file and add the following statement:

```
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values (‘SwitchA.hp.com[ 0 [ 2 ]
]’,’SwitchB.hp.com[ 0 [ 42 ] ]’,1);
```

Notice that the value for the m_Command field of the insert statement is 1 for add.

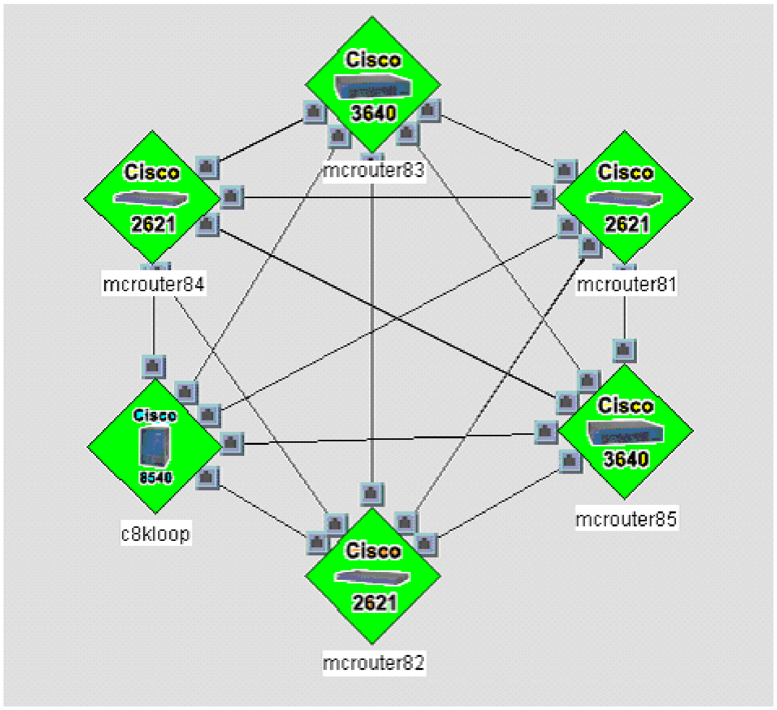
Add Connections for Unsupported Devices

Extended Topology does not yet support all network devices. When an unsupported device plays a key role in connectivity, you may see a "fully-meshed" or "false mesh" layout that does not reflect reality.

For example, the layout shown in Figure 8 on page 66 may be due to an unsupported switch that physically links each of the routers in a star configuration. Although the connectivity it provides was discovered, the device itself was not, so it is missing from the layout. You will want to use the connectionEdits file to create a correct topology model.

In Figure 8 on page 66, the false mesh may be due to an unsupported switch. Although it is physically linking each of the routers, it is missing from the center of the web in the view.

Figure 8 A False Mesh of Connections



In this situation, each of the connections on devices around the edge will be connected to other devices in the false mesh by the same interface. This means that the ifIndex will be the same as the other devices in the mesh. To remove the false mesh, you may have to insert a *delete connection* entry to the connectionEdits file for each of these connections. You will also have to insert an *add connection* entry in the connectionEdits file for each of the actual connections from the unsupported device to the switch. Use the following procedure to fix the problem in this example:

1 To create the *delete connection* entries, find the ifIndex value for each device in the false mesh. This is most easily done from a Neighbor View by moving your mouse over each port icon and noting the Interface Number field located in the popup. For the above picture, this would yield entries such as:

```
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter83.hp.com[ 0 [
3 ] ]','mcrouter81.hp.com[ 0 [ 4 ] ]',2);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter83.hp.com[ 0 [
3 ] ]','mcrouter85.hp.com[ 0 [ 6 ] ]',2);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter83.hp.com[ 0 [
3 ] ]','mcrouter82.hp.com[ 0 [ 5 ] ]',2);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter83.hp.com[ 0 [
3 ] ]','c8kloop.hp.com[ 0 [ 1 ] ]',2);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter83.hp.com[ 0 [
3 ] ]','mcrouter84.hp.com[ 0 [ 7 ] ]',2);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter81.hp.com[ 0 [
4 ] ]','mcrouter85.hp.com[ 0 [ 6 ] ]',2);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter81.hp.com[ 0 [
4 ] ]','mcrouter82.hp.com[ 0 [ 5 ] ]',2);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter81.hp.com[ 0 [
4 ] ]','c8kloop.hp.com[ 0 [ 1 ] ]',2);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter81.hp.com[ 0 [
4 ] ]','mcrouter84.hp.com[ 0 [ 7 ] ]',2);
:
:
```

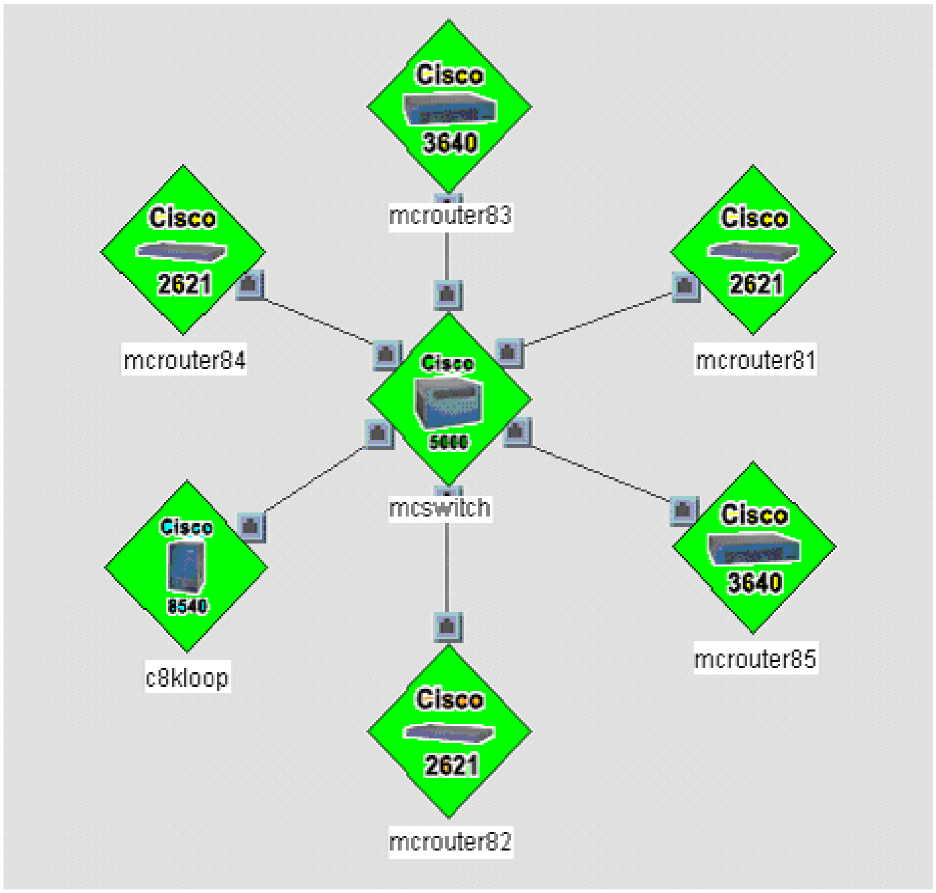
2 To create the *add connection* entries, find the ifIndex value for each of the interfaces on the unsupported device. This could be done by physically examining the device, by querying the appropriate SNMP MIB which provides forwarding table information (perhaps using the NNM MIB Application Builder tool), or by some other proprietary means.

After you obtain this connectivity information, create entries in the connectionEdits file for each of the devices located at the edge of the spider web. These entries describe their connections to the unsupported device that is located in the middle of the false mesh. This device is identified as mcswitch in the example below.

```
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter83.hp.com[ 0 [
3 ] ]','mcswitch.hp.com[ 0 [ 21 ] ]',1);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter81.hp.com[ 0 [
4 ] ]','mcswitch.hp.com[ 0 [ 22 ] ]',1);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter82.hp.com[ 0 [
5 ] ]','mcswitch.hp.com[ 0 [ 23 ] ]',1);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter84.hp.com[ 0 [
7 ] ]','mcswitch.hp.com[ 0 [ 24 ] ]',1);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('c8kloop.hp.com[ 0 [ 1 ]
]','mcswitch.hp.com[ 0 [ 25 ] ]',1);
insert into disco.connectionEdits
(m_Name,m_NbrName,m_Command) values ('mcrouter85.hp.com[ 0 [
6 ] ]','mcswitch.hp.com[ 0 [ 26 ] ]',1);
```

An Extended Topology discovery using the above additions to the connectionEdits file would result in the Neighbor View shown in Figure 9.

Figure 9 Corrected Neighbor View



Other Limitations and Behaviors

- Interfaces described in the connectionEdits file must physically exist. Extended Topology must be able to discover these interfaces.
- If you create a connection for an interface in the connectonEdits file, that does not prevent Extended Topology from creating other connections to that interface.
- If you used the ovet_topoconnedit.ovpl script to make changes to the Extended Topology database and want to see the changes you made to the connectionEdits file reflected in Dynamic Views, you must restart the

OpenView application server (ovas) and the Active Problem Analyzer (ovet_poll) processes. See Stop and Restart Processes on page 17 for more information.

- To make sure the Dynamic Views contain all of your changes, you should initiate a new discovery each time you make changes to the connectionEdits file.

  A new discovery ensures that all Extended Topology data is consistent with the changes introduced in the connectionEdits file.

Filter Interface Discovery

Interface discovery filters provide a way to limit interface discovery. There are two ways to use this functionality:

- Choose not to add information on specified interfaces to the topology databases.
- Choose to stop discovering additional interfaces on a specified node after a certain interface has been discovered ("truncate" mode).

Both actions reduce the amount of topology data that Extended Topology stitchers must analyze. Additionally, the second mechanism reduces discovery data collection.

When you limit interface discovery, those excluded interfaces will be missing from NNM topology. For example, excluded interfaces will not be listed in Node Details views. To discover interfaces but not poll them, use APA polling filters or suppress polling with the ovet_toposet command. Note that unconnected interfaces (interfaces that are not connected to any managed nodes) are, by default, discovered but not polled.

➤ You can allow or suppress filtering from any dynamic view. See *Allow or Suppress Monitoring* in the online help for more information.

Following is information you should know when you configure interface discovery filtering:

- Devices are identified by management IP address, management IP address wildcard, exact hostname, or hostname wildcards.
- Filters can be inclusive or exclusive.
- Filter behavior can truncate the interface table or remove individual interfaces.
- Fields can be tested for single or multiple integer or string values, based on the field being tested.
- Configuration is applied to both NNM Standard Edition and NNM Advance Edition with the same file:

```
$OV_CONF/netmon.interfaceNoDiscover.
```

Configuration

The file, `$OV_CONF/netmon.interfaceNoDiscover`, is used to restrict the set of interfaces to be discovered for a node. The entries in the file are applied in a sequential fashion and are terminated when the first match with the hostname column is found. The following types of lines may be in the file:

- Blank lines
- Comments
- Three columns displaying the node specification, filtering mode, and interface identification.

Following are examples of the netmon.interfaceNoDiscover file:

```
# For the node with the management address 10.162.191.146,
# truncate the interface table when an interface with ifType
# of 135 is found:
# 10.162.191.146   1   ifType=135

# For the node with name node-sw11, truncate the interface
# table when an interface with ifType of 53 is found:
# node-sw11   1   ifType=53

# For nodes with the management address in the range
# 10.162.191.0 through 10.162.191.255, ignore the interfaces
# with ifType of 135 or 53:
# 10.162.191.*   2   ifType=135,53

# For nodes with a name ending in "core", ignore the
# interfaces with ifDescr containing either the string "VLAN"
# or "virtual":
# *core   2   ifDescr="*VLAN*","*virtual*"

# For nodes with the management address in the range
# 10.162.191.10 through 10.162.191.10-19, ignore the
# interfaces with ifType not equal to 6
# 10.162.191.10-19   2   ifType!=6

# For the nodes with a name starting with "switch", truncate
# the interfaces table when an interface with ifName that does
# not start with either "Gi*" or "Fa*" is found:
# switch*   1   ifName!="Gi*","Fa*"
```

Node Specification

The node specification may contain the following:

- Exact management IP address consisting of 4 dotted decimal numbers with each number in the range of 0 to 255, for example, 192.168.1.1.
- Management IP wildcards consisting of 4 dotted decimal numbers. These can be specified as ranges (for example, 1-128) or as “*” (equivalent to 0-255). Both types can be used in the same specification, for example, 192.168.10-20.*.
- Host name wildcards (see details below)
- Exact host name matches (fully qualified if necessary)

If there are multiple matching entries, the first match is chosen.

Individual nodes are tested against this configuration using the following precedence:

1. Exact management IP address
2. Exact host name
3. IP Wildcard in the order listed in the file
4. Hostname wildcard in the order listed in the file

Filtering Mode

The filtering mode can be one of 2 values:

- 1 means to truncate the remainder of the table when the first interface matching the interface identification is encountered.
- 2 means to filter any interface matching the interface identification.

Interface Identification

The interface identification contains a field name, a comparison operation, and either a single or comma-separated list of values appropriate for the field.

- Field name - supported field names are the following:
  - ifType

    numeric field
  - ifDescr

    character field
  - ifAdminStatus

    numeric field
  - ifName

    character field
  - ifAlias

    character field

  The ifType, ifDescr, and ifAdminStatus fields and their possible values are defined in the MIB-II RFC (http://www.ietf.org/rfc/rfc1213.txt).

  The ifName and ifAlias fields are defined in the Interfaces Group MIB RFC (http://www.ietf.org/rfc/rfc2233.txt).
- Comparison operator - supported operators are the following:
  - "="

    This is the "filter out" mode of matching, that is, any interfaces found to match the supplied values will be filtered out by discovery.
  - "!="

    This is the "filter in" mode of matching, that is, only interfaces found to match the supplied values will be included by discovery.
- Values
  - Numeric fields can contain a single integer or a list of integers separated by commas. No spaces are allowed.

— Character fields can contain a single character string or a list of character strings separated by commas. Each character string must be enclosed in double quotes and may contain spaces and wildcard matching characters such as an asterisk. See information on host name wildcards below for other wildcard characters. No spaces are allowed outside of the quoted strings. Example: "VLAN*","*connection to*".

Host name wildcards may contain:

- "*"

  Use an asterisk to represent any number of characters up to the next period. For example, *.corp.com matches pc.corp.com.

- "?"

  Use a question mark to match a single character. For example, pc.c?.com matches pc.ca.com but not pc.c.com nor pc.cal.com.

- "[…]"

  Use brackets to match a single character, characters in a range, or characters not within a range if '!' is the first character within the brackets. For example: [bf]an.corp.com matches ban.corp.com and fan.corp.com; [b-d]an.corp.com matches ban.copr.com, can.corp.com and dan.corp.com; and [!c-z]an.corp.com matches only aan.corp.com and ban.corp.com.

IP Address Filters

NNM will process `$OV_CONF/netmon.noDiscover` to restrict the addresses to be discovered. See the reference page for *netmon.noDiscover* for usage information. Details about how to access reference pages are in the online help.

Validate the Filter Configuration

The filtering configuration can be validated with the $OV_BIN/etFilterConvert.ovpl support tool as well as by visual inspection.

- The -test command line option triggers basic syntax validation of $OV_CONF/netmon.interfaceNoDiscover and netmon.noDiscover.

- When the -test option is not specified, the XML filter specification used by Extended Topology for limiting and filtering SNMP requests is created. This script is run automatically by ovet_bridge prior to each discovery. The output of this script is $OV_CONF/nnmet/etconfigFilter.xml, which can also be inspected to validate correct configuration.

  New netmon action, ‘netmon –a 117’, shows the result of how netmon parses the netmon.noDiscover and netmon.interfaceNoDiscover files. Output goes to $OV_LOG/netmon.trace.

Information for Non-SNMP Nodes

You can provide limited information about non SNMP enabled devices such as `SysObjectID`, `sysLocation`, `sysContact`, and other user configurable data fields. This information is stored in the Extended Topology database.

Non-SNMP node related information can be a part of Extended Topology when you populate the following file:

UNIX:

```
$OV_CONF/nnmet/nonsnmpnodes.nnm
```

Windows:

```
%OV_CONF%\nnmet\nonsnmpnodes.nnm
```

The format and restrictions on the contents of this file are listed below:

1. There should be one row for each non-SNMP node.
2. Values in a row are separated by a comma.
3. There must be exactly 10 fields or columns in the following order:
   - a IP Address
   - b Physical Address

     Format: Six, 2 byte hexadecimal fields separated by ":" (colon)

     For example, 00:0A:0B:0C:0D:0E
   - c SysName
   - d SysContact
   - e SysObjectID (oid)

     Format: for example, 1.3.6.1.4.1.11.2.3.16.1
   - f User Definable 1
   - g User Definable 2
   - h User Definable 3
   - i User Definable 4

j User Definable 5

> Fields c, d, and f through j can hold any printable characters with the exception of commas and single quotes.

4 User definable fields can be any string that you want to populate. If a field does not have a value, leave it blank. Make sure that there are no spaces between the delimiters. If there are not 10 columns, the line is ignored.

5 None of the values can contain a comma. A comma cannot be escaped.

6 Duplicate IP addresses are not supported. Each entry must have a unique IP address.

7 Field values cannot contain a single quote. A single quote cannot be escaped.

8 After each update or addition to this file, run a zonal or full discovery to activate the new values.

9 A “#” at the beginning of any line indicates that the line is a comment, and it is ignored.

A “#” anywhere else in a line allows the part before the “#” to process, and the remainder after the “#” is ignored.

Check Syntax

Use the following script to check the syntax of this file:

```
/opt/OV/support/NM/testnonSnmpFile.ovpl
```

Run the following command to use the script:

```
testnonSnmpFile.ovpl <nonsnmpnodes.nnm>
```

This script tests each line and creates a new file in the current directory called `nonsnmpnodes.new`. In this file, all lines with errors are commented with a tag to indicate the type of error. The script displays each error on the console.

sysObjectIDs to Use

If you want to associate symbols or icons for the non-SNMP devices, use the following file:

UNIX:

```
$OV_CONF/oid_to_sym
```

Windows:

```
%OV_CONF%\oid_to_sym
```

The sysObjectIDs for non-SNMP devices must be created under the following OID:

```
1.3.6.1.4.1.11.2.3.16
```

Example Entries

Following are sample entries for a `nonsnmpnodes.nnm` file:

```
10.10.1.2,00:0A:0B:0C:0D:0E,My Box,My
Office,1.3.6.1.4.1.11.2.3.16.1,Def1,Def2,Def3,Def4,Def5


10.10.1.3,00:0A:0B:0C:0D:0A,His Box,His
Office,1.3.6.1.4.1.11.2.3.16.2,,Def2,Def3,Def4,Def5


10.10.1.4,00:0A:0B:0C:0D:0B,His
Box,,1.3.6.1.4.1.11.2.3.16.1,,,Def3,Def4,Def5
```

The sample entries are displayed on two lines due to line-length restrictions in the manual. Each entry must be on a single line in the file.

Troubleshooting

- Problem:

  `ovet_disco` is stuck or hangs at “initializing” state.

  Possible Solution:

  Determine if the `nonsnmpnodes.nnm` file exists. You can find the file at the following location:

  UNIX:

  ```
  $OV_CONF/nnmet
  ```

  Windows:

  ```
  %OV_CONF%\nnmet
  ```

  If the file does not exist then create an empty file in the directory.

- Problem:

  The icons associated with the specified OID do not show up in the Node Views (or other Dynamic Views).

  Possible Solution:

  Complete the following actions:

  — Run the command the following command:

  ```
  ovdvstylereset
  ```

  — Stop and restart the `ovas` process. See Stop and Restart Processes on page 17 for more information.

- Problem:

  The information populated in the `nonsnmpnodes.nnm` file is not getting populated in the ET DB.

  Possible Solutions:

  — Check for syntax errors. See Check Syntax on page 78 for more information.

  — Ensure a successful new discovery has been completed with the following command:

  ```
  testnonSnmpFile.ovpl <nonsnmpnodes.nnm>
  ```

Retry and Timeout Values From Classic NNM

Retry and timeout configuration is specified with host names, IP addresses, and IP address wildcards in classic NNM. You can configure Extended Topology discovery to import these retry and timeout values from classic NNM.

By default, this option is disabled.

To enable the option to import retry and timeout values, complete the following steps:

1. Run the following command to stop the `ovet_dhsnmp` process:

   ```
   ovstop -c ovet_dhsnmp
   ```

2. Open the following file for editing:

   - *UNIX*:

     ```
     $OV_LRF/ovet_dhsnmp.lrf
     ```

   - *Windows*:

     ```
     %OV_LRF%\ovet_dhsnmp.lrf
     ```

3. Find the following line in the `ovet_dhsnmp.lrf` file:

   ```
   OVs_YES_START:ovet_dhelpserv::OVs_WELL_BEHAVED:30:NOPAUSE
   ```

4. Add the bolded text to the line as follows:

   ```
   OVs_YES_START:ovet_dhelpserv:-useNNMSNMPConf:OVs_WELL_BEH
   AVED:30:NOPAUSE
   ```

5. To activate your changes, run the following command:

   ```
   $OV_BIN/ovaddobj ovet_dhsnmp.lrf
   ```

6. Run the following command to start the `ovet_dhsnmp` process:

   ```
   ovstart -c ovet_dhsnmp
   ```

Community String Automatic Discovery

By default, Extended Topology exports and applies community strings from the set of community strings that are assigned in classic NNM. If you do not choose to export community strings from classic NNM, Extended Topology discovery chooses a community string that best matches each device.

To disable the automatic discovery of community strings, complete the following steps:

1. Run the following command to stop the `ovet_dhsnmp` process:

   ```
   ovstop -c ovet_dhsnmp
   ```

2. Open the following file for editing:
   - *UNIX*:

     ```
     $OV_LRF/ovet_dhsnmp.lrf
     ```

   - *Windows*:

     ```
     %OV_LRF%\ovet_dhsnmp.lrf
     ```

3. Find the following line in the `ovet_dhsnmp.lrf` file:

   ```
   OVs_YES_START:ovet_dhelpserv::OVs_WELL_BEHAVED:30:NOPAUSE
   ```

4. Add the bolded text to the line as follows:

   ```
   OVs_YES_START:ovet_dhelpserv:-noAutoDiscoverCmStr:
   OVs_WELL_BEHAVED:30:NOPAUSE
   ```

5. To activate your changes, run the following command:

   ```
   $OV_BIN/ovaddobj ovet_dhsnmp.lrf
   ```

6. Run the following command to start the `ovet_dhsnmp` process:

   ```
   ovstart -c ovet_dhsnmp
   ```

Cisco Discovery Configuration

Cisco Discovery Configuration allows network management administrators to control how much detail the Extended Topology discovery process collects for certain Cisco network devices.

Extended Topology discovery gathers a rich set of information for Cisco network devices. While this information is useful for managing these devices in day-to-day operations, not all users need the full set of data. For very large, managed environments, an Extended Topology discovery can take a long time.

CDC allows network administrators to make tradeoffs between discovering more information and discovering the network environment more quickly. If certain types of detailed information are not needed, collection of that data can be eliminated. This action can shorten discovery times, often significantly.

For more information on this advanced feature, see the Cisco Discovery Configuration whitepaper at the following location:

- *UNIX*:

  ```
  /opt/OV/doc/WhitePapers
  ```

- *Windows*:

  ```
  \Program Files\HP OpenView\doc\WhitePapers
  ```

3 Dynamic Views

NNM provides numerous views that use Extended Topology information to help you monitor and troubleshoot your network environment. All the views are accessed through the Home Base user interface. See Launch Home Base on page 86 for more information.

The views present a graphical or tabular representation of your network infrastructure. These views are based on criteria in effect at the time a request is made. Views based on Extended Topology or enhanced by Extended Topology include the following views:

- Container View
- Node Status and Interface Status Views
- Alarm View
- Neighbor View
- Node View
- Interface View
- Path View
- VLAN View
- Problem Diagnosis View
- HSRP View
- VRRP View
- Overlapping Address Domain View
- OSPF View

The Dynamic Views contain tools such as Ping and Trace Route that originate from the system you use to launch the browser. Access controls and security restrictions for these commands are based on the rules that apply to the system and user of the browser.

Launch Home Base

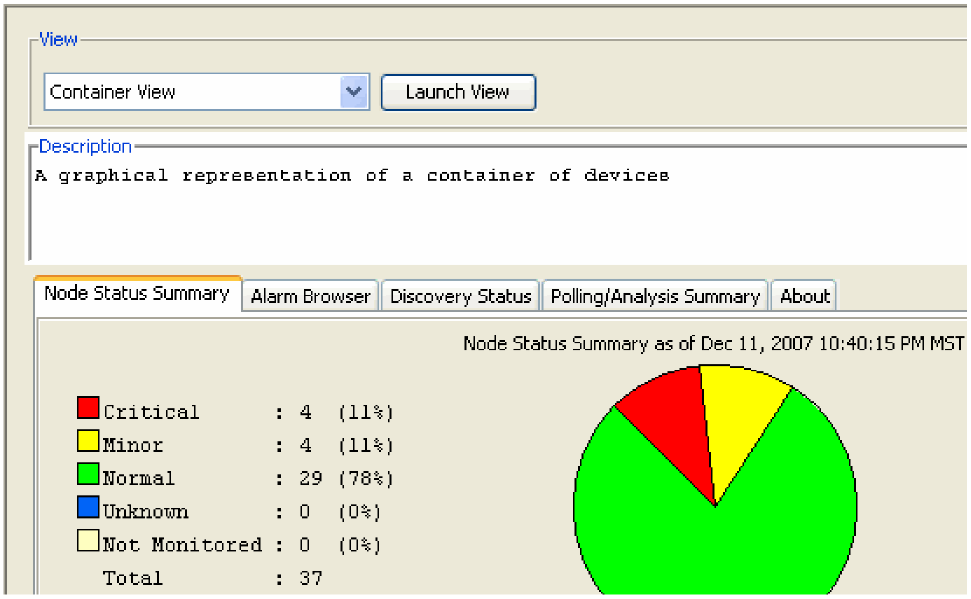
There are two ways to launch Home Base and access Dynamic Views - as an applet and as an application.

- **Applet** - You must use a supported web browser and have a correct Java Plug-In (JPI) installed to access Extended Topology views with the applet. See Access Dynamic Views as an Applet on page 87 for more information.
- **Application** - You can install Dynamic Views on your Windows client system, using Java Web Start. With this option, you must have a correct Java Runtime Environment (JRE) installed, but the JPI is not used. See Access Dynamic Views as an Application on page 87 for more information.

For information about supported web browsers and JRE/JPI versions, see the Release Notes.

Home Base is a launching point for Dynamic Views. See Figure 10 for an example of Home Base. In addition to launching views from Home Base, you can select tabs that allow you to configure NNM and display additional information about your network.

Figure 10 Home Base



```
A graphical representation of a container of devices
```

Node Status Summary as of Dec 11, 2007 10:40:15 PM MST

```
🟥 Critical      : 4  (11%)
🟨 Minor         : 4  (11%)
🟩 Normal        : 29 (78%)
🟦 Unknown       : 0  (0%)
⬜ Not Monitored : 0  (0%)
   Total         : 37
```

Access Dynamic Views as an Applet

You can access Home Base from your browser using the following URL:

```
http://<hostname>:7510
```

For more information on the features available for Home Base, see the online help.

Access Dynamic Views as an Application

This application is supported only on clients that run a supported Windows operating system and use Microsoft Internet Explorer version 6.0.2900 or higher.

You can install Windows-based clients with Java Web Start (JWS). If you choose to launch Dynamic Views from the JWS application, it does not change your ability to use the applet. The URL for the applet continues to work as expected.

To install the JWS Dynamic Views application, you need to launch a URL and follow the installation steps. See Install the Java Web Start Application on page 87 for more information.

The installation process provides a shortcut on your desktop if you select that option. Otherwise, launch the JWS application from the Java Web Start program in your Windows All Programs list. Select **Network Node Manager Home Base** and then select **Start** to start the Dynamic Views User Interface.

Install the Java Web Start Application

To install the JWS application, complete the following steps:

1. Enter the following URL in your browser to launch the NNM 7.5 Dynamic Views Client Installer.

   ```
   http://<NNM_ServerName>:7510/topology/webstart/index.html
   ```

2. Select **Install and Launch Home Base** from the installer page.
3. Follow any prompts on the installation screens. For best results, accept the defaults.
4. When installation is complete, the JWS Home Base launches.

Remove the Java Web Start Application

The removal procedure is different for the two supported Java Runtime Environments (JRE).

To remove the JRE 1.4 Java Web Start application, complete the following steps:

1 Close all web browsers.

2 Select the Java Web Start program in your Windows All Programs list.

3 Select **Network Node Manager Home Base**.

4 Select the Application menu and click **Remove Application**.

To remove the JRE 1.5 Java Web Start application, complete the following steps:

1 Close all web browsers.

2 Select **Start** → **Control Panel** → **Java** to launch Java Control Panel.

3 On the **General** tab, select the **Settings** button for the Temporary Internet Files.

4 Select the **View Applications** button to open the Java Application Cache Viewer.

5 In the table, select **Network Node Manager Home Base**, and click **Remove**.

Configure Dynamic Views

You can modify parameters to customize your Dynamic Views experience. This section contains information about parameter changes that can improve the usability performance of Dynamic Views in your environment. The following two files are used to customize Dynamic Views:

- web.xml File
- dynamicViews.conf File

web.xml File

The web.xml file resides at the following location:

- *UNIX*:

  `$OV_AS/webapps/topology/WEB-INF/web.xml`

- *Windows*:

  `%OV_AS%\webapps\topology\WEB-INF\web.xml`

You can make changes directly to the web.xml file, but it is better practice to modify the smaller XML files in the `WEB-INF` directory. These files are combined to build the web.xml file. With the smaller files, you can make focused changes without the risk of introducing errors to the `web.xml` file. It is also easier to manage your changes when you install patches or upgrade to a newer version.

➤ Before you make changes to this file or any other configuration file, make a backup copy. A backup copy allows you to recover from errors you might inadvertently introduce.

Apply Parameter Changes to web.xml

After you modify one of the XML files in the `WEB-INF` directory, run the following script to add your changes to the `web.xml` file.

- *UNIX*:

  ```
  $OV_NEW_CONF/OVNNM-RUN/www/registration/dynamicViews/
  oneXmlFileCreator/oneXmlFileCreator.ovpl
  ```

- *Windows*:

  ```
  %OV_MAIN_PATH%\newconfig\OVNNM-RUN\www\registration\
  dynamicViews\oneXmlFileCreator\oneXmlFileCreator.ovpl
  ```

dynamicViews.conf File

The dynamicViews.conf file resides at the following location:

- *UNIX*:

  ```
  $OV_CONF/dynamicViews.conf
  ```

- *Windows*:

  ```
  %OV_CONF%\dynamicViews.conf
  ```

➤ Before you make changes to this file or any other configuration file, make a backup copy. A backup copy allows you to recover if errors you cannot find result from your changes.

Dynamic Views Configuration

There are many parameter changes you can make. The changes noted in this section are those changes that have the potential to make a difference in your environment.

Change Interface Name Format in Dynamic Views

By default, ifIndex and board number are used to display interface names in Node Details and Interface Details views. For example, mynode[0 [4]] represents the following:

- Node name is mynode
- Board number is 0
- ifIndex is 4

You can change the way interface names are displayed. The alternate way to display interfaces is to display the ifName. You might want to use the ifName for interface names if your environment primarily consists of Cisco devices or if you are simply more familiar identifying an interface by its ifName.

To configure Dynamic Views to display interface names based on ifName, complete the following steps:

1 Create a backup copy of the following file:

   - *UNIX*:

     ```
     $OV_CONF/dynamicViews.conf
     ```

   - *Windows*:

     ```
     %OV_CONF%\dynamicViews.conf
     ```

2 Open the `dynamicViews.conf` file for editing, and find the following block of text:

   ```
   enableDisplayIfName false
   ```

3 Change the bolded value from false to true to display interface names based on ifName.

4 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

   ➤ If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Disable Ping Information Display on Node Details Page

By default, ping information displays on the Node Details page. This can be a problem if APA is not enabled to poll IPv4 nodes. In this case, ping information for nodes comes from netmon, and this information can be different from the interface ping status. To remove the potentially conflicting information, you can turn off the ping information on the Node Details page.

To configure Dynamic Views to not display ping information on the Node Details page, complete the following steps:

1 Create a backup copy of the following file:

   - *UNIX*:

     ```
     $OV_CONF/dynamicViews.conf
     ```

   - *Windows*:

     ```
     %OV_CONF%\dynamicViews.conf
     ```

2 Open the `dynamicViews.conf` file for editing, and find the following block of text:

```
disablePingInfo false
```

3 Change the bolded value from false to true to disable ping information on the Node Details page.

4 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

➤ If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Change the Maximum Number of Nodes in Neighbor View

You can change the default maximum number of nodes that Neighbor View displays in a single view. If you make this value too large, the view performance will degrade so use caution as you change this parameter. Adequate performance of Neighbor Views cannot be guaranteed if this value is increased above the default.

To change the maximum number of nodes in Neighbor Views, complete the following steps:

1 Create a backup copy of the following file:

- *UNIX*:

  ```
  $OV_AS/webapps/topology/WEB-INF/servletRegistration/
  1NeighborServlet.xml
  ```

- *Windows*:

  ```
  %OV_AS%\webapps\topology\WEB-INF\servletRegistration\
  1NeighborServlet.xml
  ```

2 Open the `1NeighborServlet.xml` file for editing, and find the following block of text:

```
<init-param>
   <param-name>maxNeighbors</param-name>
   <param-value>200</param-value>
</init-param>
```

3 Change the bolded value to the maximum number of nodes you want to display by default in Neighbor Views.

4 After your changes are complete, run the script to add your changes to the web.xml file. See Apply Parameter Changes to web.xml on page 89 for more information.

5 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

➤ If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Set the Default Hop Count in Neighbor View

By default, the Neighbor View hop count is set at 2. You can adjust this value to a number between 1 and 9. In a complex network, we recommend you set the hop count to 1 to improve the performance and usability of your Neighbor Views.

To change the default hop count in Neighbor Views, complete the following steps:

1 Create a backup copy of the following file:

- *UNIX*:

```
$OV_AS/webapps/topology/WEB-INF/servletRegistration/
1NeighborServlet.xml
```

- *Windows*:

```
%OV_AS%\webapps\topology\WEB-INF\servletRegistration\
1NeighborServlet.xml
```

2 Open the `1NeighborServlet.xml` file for editing, and find the following block of text:

```
<init-param>
   <param-name>defaultNumHops</param-name>
   <param-value>2</param-value>
</init-param>
```

3 Change the bolded value to the number hops that is best suited to your environment.

4 After your changes are complete, run the script to add your changes to the web.xml file. See Apply Parameter Changes to web.xml on page 89 for more information.

5 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

> If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Set the Maximum Hop Count in Neighbor View

By default, the Neighbor View maximum number of hops is set at 9. You can adjust this value to a number between 1 and 9.

> The value for maximum node count in a Neighbor View is also configurable. Regardless of how high you set the hop count, Neighbor View will not display more nodes than the configured maximum node count.

The maximum node count is set to a certain value regardless of how many hops you choose, you won't get more than max node count

To change the default hop count in Neighbor Views, complete the following steps:

1 Create a backup copy of the following file:

- *UNIX*:

```
$OV_AS/webapps/topology/WEB-INF/servletRegistration/
1NeighborServlet.xml
```

- *Windows*:

```
%OV_AS%\webapps\topology\WEB-INF\servletRegistration\
1NeighborServlet.xml
```

2 Open the `1NeighborServlet.xml` file for editing, and find the following block of text:

```
<init-param>
   <param-name>maxNumHops</param-name>
   <param-value>9</param-value>
</init-param>
```

3 Change the bolded value to the maximum number hops that is best suited to your environment.

4 After your changes are complete, run the script to add your changes to the web.xml file. See Apply Parameter Changes to web.xml on page 89 for more information.

5 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information

➤ If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Use Only Extended Topology Data in Neighbor View

By default, the Neighbor View uses data from classic NNM. You can change the default behavior for the Neighbor View so it uses only Extended Topology data.

To configure Neighbor View to use only Extended Topology data, complete the following steps:

1 Create a backup copy of the following file:

- *UNIX*:

  ```
  $OV_AS/webapps/topology/WEB-INF/servletRegistration/
  1NeighborServlet.xml
  ```

- *Windows*:

  ```
  %OV_AS%\webapps\topology\WEB-INF\servletRegistration\
  1NeighborServlet.xml
  ```

2 Open the `1NeighborServlet.xml` file for editing, and find the following block of text:

```
<init-param>
   <param-name>restrictToET</param-name>
   <param-value>0</param-value>
</init-param>
```

3 Change the bolded value from 0 to 1 to use only Extended Topology data in Neighbor Views.

4 After your changes are complete, run the script to add your changes to the web.xml file. See Apply Parameter Changes to web.xml on page 89 for more information.

5 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

➤ If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Restrict Neighbor View to One Hop with Classic NNM Starting Node

This option places a 1 hop count limit for Neighbor Views with a starting node that is only found in the classic NNM database. An example is a Neighbor View for an end node that is not discovered by Extended Topology because you have specified this restriction for this end node.

If a multiple hop Neighbor View is initiated with a classic NNM starting node, ovas process performance can degrade and create performance problems in your Dynamic Views. When you set this option, operators cannot initiate a multiple hop count Neighbor View from a starting node found only in the classic NNM database. This change is strongly recommended for large managed networks.

To enable this option in Neighbor Views, complete the following steps:

1 Create a backup copy of the following file:

- *UNIX*:

  ```
  $OV_AS/webapps/topology/WEB-INF/servletRegistration/
  1NeighborServlet.xml
  ```

- *Windows*:

  ```
  %OV_AS%\webapps\topology\WEB-INF\servletRegistration\
  1NeighborServlet.xml
  ```

2 Open the `1NeighborServlet.xml` file for editing, and find the following block of text:

```
<init-param>
   <param-name>switchToTopmdWithOneHop</param-name>
   <param-value>0</param-value>
</init-param>
```

3 Change the bolded value from 0 to 1.

4 After your changes are complete, run the script to add your changes to the web.xml file. See Apply Parameter Changes to web.xml on page 89 for more information.

5 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

➤ If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Change the Maximum Number of Nodes in Node View

You can change the default maximum number of nodes that Node View displays in a single view. If you make this value too large, the view performance will degrade so use caution as you change this parameter. Adequate performance of Node Views cannot be guaranteed if this value is increased above the default.

To change the maximum number of nodes in Node View, complete the following steps:

1. Create a backup copy of the following file:
   - *UNIX*:

     ```
     $OV_AS/webapps/topology/WEB-INF/servletRegistration/
     1NodeServlet.xml
     ```

   - *Windows*:

     ```
     %OV_AS%\webapps\topology\WEB-INF\servletRegistration\
     1NodeServlet.xml
     ```

2. Open the `1NodeServlet.xml` file for editing, and find the following block of text:

   ```
   <init-param>
       <param-name>maxNodes</param-name>
       <param-value>250</param-value>
   </init-param>
   ```

3. Change the bolded value to the maximum number of nodes you want to display by default in Node Views.
4. After your changes are complete, run the script to add your changes to the web.xml file. See Apply Parameter Changes to web.xml on page 89 for more information.
5. Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

   ➤ If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Use Extended Topology Filters in Node View

When you manage larger networks with Extended Topology, the default set of classic NNM filters in Node View can result in slow performance. To significantly improve the performance of Node View, you can configure Node View to use Extended Topology Filters. See Node View on page 119 for more information.

By default, Extended Topology filters are used in the Container View. If you make this change, the behavior of the views will be more consistent.

To enable or disable Extended Topology filters in Node View, complete the following steps:

1. Create a backup copy of the following file:
   - *UNIX*:

     ```
     $OV_AS/webapps/topology/WEB-INF/servletRegistration/
     1NodeServlet.xml
     ```

   - *Windows*:

     ```
     %OV_AS%\webapps\topology\WEB-INF\servletRegistration\
     1NodeServlet.xml
     ```

2. Open the `1NodeServlet.xml` file for editing, and find the following block of text:

   ```
   <init-param>
      <param-name>enableETFilters</param-name>
      <param-value>1</param-value>
   </init-param>
   ```

   If this block of text does not exist, add it **below** the following block:

   ```
   <init-param>
      <param-name>maxNodes</param-name>
      <param-value>250</param-value>
   </init-param>
   ```

3. The value of 1 in the block of text in the above step enables Extended Topology filters. To disable Extended Topology filters, change the value to 0.

4. After your changes are complete, run the script to add your changes to the web.xml file. See Apply Parameter Changes to web.xml on page 89 for more information.

5 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

> If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Change the Maximum Number of Interfaces in Interface View

You can change the default maximum number of interfaces that Interface View displays in single view. If you make this value too large, the view performance will degrade so use caution as you change this parameter. Adequate performance of Interface Views cannot be guaranteed if this value is increased above the default.

To change the maximum number of interfaces in Interface Views, complete the following steps:

1 Create a backup copy of the following file:

- *UNIX*:

```
$OV_AS/webapps/topology/WEB-INF/servletRegistration/
1IfaceServlet.xml
```

- *Windows*:

```
%OV_AS%\webapps\topology\WEB-INF\servletRegistration\
1IfaceServlet.xml
```

2 Open the `1IfaceServlet.xml` file for editing, and find the following block of text:

```
<init-param>
   <param-name>maxIfaces</param-name>
   <param-value>250</param-value>
</init-param>
```

3 Change the bolded value to the maximum number of interfaces you want to display by default in Interface Views.

4 After your changes are complete, run the script to add your changes to the web.xml file. See Apply Parameter Changes to web.xml on page 89 for more information.

5 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

> If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Remove Disabled Interfaces from Interface View

By default, the Interface View displays disabled interfaces. You can change the default behavior to not display disabled interfaces.

To configure Interface View to not display disabled interfaces, complete the following steps:

1 Create a backup copy of the following file:

- *UNIX*:

```
$OV_AS/webapps/topology/WEB-INF/servletRegistration/
1IfaceServlet.xml
```

- *Windows*:

```
%OV_AS%\webapps\topology\WEB-INF\servletRegistration\
1IfaceServlet.xml
```

2 Open the `1IfaceServlet.xml` file for editing, and find the following block of text:

```
<init-param>
   <param-name>includeDisabled</param-name>
   <param-value>1</param-value>
</init-param>
```

3 Change the bolded value from 1 to 0 so disabled interfaces are not displayed.

4 After your changes are complete, run the script to add your changes to the web.xml file. See Apply Parameter Changes to web.xml on page 89 for more information.

5 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

> If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Change the Maximum Number of Status Entries in Interface Status View

You can change the default maximum number of interfaces status entries that Interface Status View displays. Adequate performance of Interface Status Views cannot be guaranteed if this value is increased above the default.

To change the maximum number of interface status entries in Interface Status Views, complete the following steps:

1 Create a backup copy of the following file:

- *UNIX*:

```
$OV_AS/webapps/topology/WEB-INF/servletRegistration/
1IfaceStatusServlet.xml
```

- *Windows*:

```
%OV_AS%\webapps\topology\WEB-INF\servletRegistration\
1IfaceStatusServlet.xml
```

2 Open the 1IfaceStatusServlet.xml file for editing, and find the following block of text:

```
<init-param>
   <param-name>maxIfaces</param-name>
   <param-value>1000</param-value>
</init-param>
```

3 Change the bolded value to the maximum number of interface status entries you want to display by default in Interface Status Views.

4 After your changes are complete, run the script to add your changes to the web.xml file. See Apply Parameter Changes to web.xml on page 89 for more information.

5 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Change the Maximum Number of Networks or Routers in Internet View

You can change the default maximum number of networks or routers that Internet View displays in a single view. Internet View stops adding networks and routers to an Internet View if either threshold is exceeded.

If you make the networks or routers value too large, the view performance will degrade so use caution as you change these parameters. Adequate performance of Internet Views cannot be guaranteed if either of these values is increased above the default.

To change the maximum number of networks or routers in Internet View, complete the following steps:

1 Create a backup copy of the following file:

- *UNIX*:

```
$OV_AS/webapps/topology/WEB-INF/servletRegistration/
1OVtopmdServlets.xml
```

- *Windows*:

```
%OV_AS%\webapps\topology\WEB-INF\servletRegistration\
1OVtopmdServlets.xml
```

2 Open the 1OVtopmdServlets.xml file for editing.

Find the following block of text to change the maximum number of networks:

```
<init-param>
   <param-name>maxNetworks</param-name>
   <param-value>150</param-value>
   <description>
      It is recommended that the value of the maxNetworks
      parameter be set to 150 for best results.
   </description>
</init-param>
```

Find the following block of text to change the maximum number of routers:

```
<init-param>
   <param-name>maxRouters</param-name>
   <param-value>100</param-value>
   <description>
      It is recommended that the value of the maxRouters
      parameter be set to 100 for best results.
   </description>
</init-param>
```

3 Change the appropriate bolded value to the maximum number of networks or routers that you want to display by default in Internet Views.

4. After your changes are complete, run the script to add your changes to the web.xml file. See Apply Parameter Changes to web.xml on page 89 for more information.
5. Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

   ➤ If you want to make more than one configuration change, make all your changes first, and then stop/restart the ovas process.

Access Dynamic Views

After Extended Topology completes a discovery, you can access Dynamic Views in the following ways:

- From Home Base (see Launch Home Base on page 86)
- From the classic NNM menu, select **Tools** → **Views**.
- From the Network Presenter menu, select **Tools** → **Views**.
- From the Launcher, select the **Tools** tab.
- From the Alarm Browser:
  a. Select an alarm from the Alarm Browser.
  b. Select **Actions** → **Views** using the Alarm Browser menus.
  c. Select any of the NNM or Extended Topology views located in the **Views** menu.

  ➤ You can configure the Alarm Browser to access any view or URL from a specific event. You cannot configure the web-based Alarm Browser to access any view or URL from a specific event.

Container View

The Container View is added with the Extended Topology functionality and provides similar capabilities as those you find in the classic NNM user interface; for example, submaps, status propagation, and background graphics. Containers are similar to submaps because they allow nodes to be grouped together into a single view. The containers are hierarchical. This means they can be created to allow drill-down navigation from one container to another. Background graphics can be applied to a container so that it displays behind the nodes in the view.

You can configure containers to show layer 2 connectivity from Extended Topology. They can also be configured to show connectivity between two nodes, between nodes and containers, and between two containers.

The top level container is called the Root Container and is the first container launched by default. Containers you create are empty until you populate them with nodes and/or other containers. You can add nodes manually, or you can associate a container with an Extended Topology filter to auto-populate the container with the set of nodes that pass the specified filter. Containers that are based on filters are auto-populated after each Extended Topology discovery cycle completes. If APA is enabled, node status is propagated up to the container, and each container's status is propagated up to the top level container. If APA is not enabled, status does not propagate up through the container hierarchy.

A typical use of containers is to divide your network into groups by physical location, such as by region, city, building, and so on. You can use hierarchy to subdivide a location into smaller regions. In this case, you would put the nodes for a smaller region inside a lower level container. The status of each container can then be used to monitor the status of each respective location.

For more information on using the Container View, see the online help.

Container View Access Controls

By default, every container is accessible to every user. Administrators can customize container access to make the Container View more effective in your environment. Before you can use Container View access control, user authentication must be enabled. See the *Extended Topology and APA Deployment Guide* in the whitepaper directory for more information on user authentication and container access controls.

Enable Container Access Control

Container access control is defined in the containers.xml file. See Container Definition File: containers.xml on page 107 and The containers Tag on page 109 for more information.

Container Setup Menu

In the Container View, use the Container Setup menu to create and modify containers. The following list describes the Container Setup menu actions:

- **Save Layout**

  Save the container layout. If you do not save the layout, it will revert to the previously saved layout or to an automatic layout if a layout for this container has never been saved.

- **Show Node Connections**

  Select this option to display connections between nodes.

- **Show Container Connections**

  Select this option to display connections between containers.

- **New Container**

  Create a new container. It is a good idea to provide a name that is descriptive of the container's contents.

- **Include Node**

  Add an existing node to a container. The node name you provide must match the Extended Topology name for the node. To review a list of node names, run the following command:

  ```
  ovet_topodump.ovpl -node
  ```

- **Include Container**

  Add an existing container to a container. Select the container you want to include from a list.

- **Exclude**

  Select the node or container you want to remove from a Container View, and then choose **Exclude**. When you remove a node or container from a container, the node or container remains in the topology.

- **Copy To Another Container**

  Select the node or container you want to copy, choose **Copy To Another Container**, and then select the target container from a list .

- **Set Container Filter**

  Select a filter for a container from the list of available filters. When you use a filter to populate a container, only nodes that pass that filter will populate the container.

- **Delete**

  Select the container you want to delete from a Container View, and choose **Delete**. The selected container is deleted, but the contents of that container remain. For example, suppose Container A contains Container B. If Container A is deleted, Container B is removed from Container A, but Container B is not deleted or changed. Container B continues to show up in the list of containers and can be included in other containers.

- **Rename**

  Select the container you want to rename, and choose **Rename**. When you rename a container, all references to that container are updated.

- **Set User Access**

  A checkbox menu lists users you can choose. Each person whose name is selected has access to the current container. Set User Access is only available if container access control has been enabled. See Enable Container Access Control on page 105 for more information.

- **Set Role Access**

  A checkbox menu lists user roles you can choose. Each user role that is selected has access to the current container. Set Role Access is only available if container access control has been enabled. See Enable Container Access Control on page 105 for more information.

Restrict Access to the Container Setup Menu

You can restrict access to this menu with the following actions:

- Modify the containers tag. See The containers Tag on page 109 for more information.
- Enable container access control. When container access control is enabled for a container, the behavior of the Container Setup menu changes. By default, the Container Setup menu is visible to every user, but when access control is enabled, only users with the role of administrator can see the menu. See the *Extended Topology and APA Deployment Guide* in the whitepaper directory for more information on container access control.

Container Definition File: containers.xml

The containers.xml file contains the complete set of container definitions. You can find this file at the following location:

- UNIX:

  ```
  $OV_DB/nnmet/containers.xml
  ```

- Windows:

  ```
  %OV_DB%\nnmet\containers.xml
  ```

The Container View provides the ability to create and maintain containers. However, there are some cases where it may be necessary to directly modify the containers.xml file. Those cases include:

- Enable user access control.
- Hide the Container Setup menu so that changes cannot be made from the Container View.
- Change the symbol icon for a container reference.
- Add a background graphics to a container.
- Restructure the overall container structure.
- Auto-generate the initial containers.xml file from a script.

Edit the Container Definition File

Do not edit the containers.xml file directly in the run-time directory. This file is read and overwritten by the ovas server during normal operation.

To edit the container definition file, complete the following steps:

1. Make a backup copy of the containers.xml file.
2. Make a separate working copy of the containers.xml file. Make any changes to the working copy only.
3. Copy the edited copy of the file back into the run-time directory when your changes are complete.
4. When a change is made to the containers.xml file, the ovas server must reload the file before the Container View reflects the change. Use one of the following ways to accomplish this:

    a. Stop and restart the ovas process. See Stop and Restart Processes on page 17 for more information.

    b. Initiate a reload using the reloadContainers option. To initiate a reload, enter the following URL into a browser:

    ```
    http://<hostname>:7510/topology/
    home?reloadContainers=true
    ```

    c. Wait until the next Extended Topology discovery cycle completes.

When you reload containers, the following actions happen:

- The following text is logged to the ovas.log file:

  ```
  Starting thread to initialize container topology ...
  ```

- When the reload is completed, the following text is logged:

  ```
  Container topology initialization thread completed
  ```

The ovas.log file is located in the following location:

- *UNIX*:

  ```
      $OV_PRIV_LOG/ovas.log
  ```

- *Windows*:

  ```
      %OV_PRIV_LOG%\ovas.log
  ```

Container Tags

This section covers the tags used in the containers.xml file.

The Root Container

The default container that appears when the Container View opens is called the Root Container. It is defined by the following tag:

```
<container name="Root Container">
```

The Root Container must always exist in the containers.xml file.

Do not confuse the Root Container with the root XML tag.

The containers Tag

The root tag in the file is as follows:

```
<containers enableContainerSetup="true">
```

The attributes of the containers tag are listed in the following table.

Attributes	Definition
enableContainerSetup	• Boolean value that is set to “true” or “false” • Defines if the Container Setup menu is shown in the Container View • Ignored if enableAccessControl is set to “true”
xmlns:xsi	• XML schema instance • Factory setting – do not change
xsi:noNamespaceSchemaLocation	• XSD schema location • Factory setting – do not change
enableAccessControl	• Boolean value that is set to “true” or “false” • Defines if container access control is enabled

The container Tag

Each container in the containers.xml file is represented as a child tag. The tag is as follows:

```
    <container name=”the container name”>
```

The attributes of the container tag are listed in the following table.

Attributes	Definition
name	• Name of the container
showNodeConnectivity	• Boolean value that is set to "true" or "false" • Defines if the view shows connections between the nodes
showContainerConnectivity	• Boolean value that is set to "true" or "false" • Defines if the view shows connections between container symbols or between nodes and container symbols

The background Tag

A background image, for example, a GIF file, can display in a Container View using the background tag. The background graphic is defined with the container's child tag as follows:

```
<background image="name of the background image file">
```

The image file must reside in the following path:

- *UNIX*:

  ```
  $OV_WWW/htdocs/images/backgrounds
  ```

- *Windows*:

  ```
  %OV_WWW%\htdocs\images\backgrounds
  ```

Create the "backgrounds" directory if it does not already exist.

The background image scales to the current zoom level of the Container View window and is proportional to the size of the node and container icons in the view. Use a graphics tool to enlarge or shrink the background image if you want to change the relative size of the background image with respect to the icons.

Supported background file format types are: GIF, JPEG, and PNG.

The topoNode tag

Each node in a container is specified by the container's child tag as follows:

```
<topoNode>
```

The attributes of the topoNode tag are listed in the following table.

Attributes	Definition
name	• Name of the node as it exists in the Extended Topology database
objId	• Unique Extended Topology object ID of the node as it exists in the Extended Topology database
x, y	• x and y coordinates of the node in the Container View

If the topoNode only has a name attribute specified, the object ID is determined when the containers are loaded, and the objId attribute is set automatically. The x and y coordinates are also set automatically when the container layout is saved.

The containerReference tag

A containerReference is a reference to another container that is in the current container. Each container reference in a container is specified by the container's child tag as follows:

```
<containerReference>
```

The attributes of the containerReference tag are listed in the following table.

Attributes	Definition
name	• Name of the container it references - the container that opens when the user double-clicks on the symbol in the Container View
symbol	• Name of the symbol icon type to display in the Container View
x, y	• x and y coordinates of the container reference symbol in the Container View

The symbol attribute allows you to select alternate symbol types. The symbol types have the same format as symbol types used in the ovw user interface, such as Container:Network.

The filter Tag

A container can be defined using an Extended Topology filter with the container's child tag as follows:

```
<filter name=”Extended Topology filter name”>
```

This tag populates the container with all the nodes that pass the specified Extended Topology filter. When the container data is loaded into the server, a topoNode tag id is created in this container for each of the nodes. Any additional nodes specified by a topoNode tag is also included in the container.

The filter is applied only when the containers.xml file is reloaded.

The accessList Tag

The accessList tag contains the list of users and/or roles that are granted access to this container. An example follows:

```
<accessList>
   <user name="myuser"/>
   <role name="myrole"/>
</accessList>
```

containers.xml Example

```
<?xml version="1.0" encoding="UTF-8"?>
<containers enableAccessControl=”true”
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="containers.xsd">
   <container name="Root Container"
   showContainerConnectivity="false">
      <containerReference name="Infrastructure Devices"
      x="635.1381324704454" y="331.28996493117836"/>
      <containerReference name="US" symbol="Container:Site"
      x="490.9852751032108" y="328.4961585859253"/>
      <containerReference name="Test Routers"
      x="558.0573994916468" y="443.05739949164683"/>
   </container>
   <container name="Infrastructure Devices"
   showNodeConnectivity="false">
      <filter name="InfrDev"/>
   </container>
   <container name="US">
      <background image="us.gif"/>
      <filter name="USRouters"/>
   </container>
   <container name="Test Routers">
      <containerReference name="My Routers"
      x="34.78907363121807" y="205.92781708671546"/>
      <topoNode name="10.2.149.12"
      objId="63c9518a-b083-71da-0885-0f0273880000"
      x="40.9" y="42.86"/>
      <topoNode name="10.2.150.140"
      objId="63e25b6c-b083-71da-0885-0f0273880000"
      x="298.0" y="434.0"/>
      <topoNode name="10.2.150.141"
      objId="63e6c3a0-b083-71da-0885-0f0273880000"
      x="616.9" y="41.86"/>
   </container>
   <container name="My Routers">
      <accessList>
         <user name="myuser" home="true"/>
      </accessList>
      <topoNode name="10.2.49.1"/>
      <topoNode name="10.2.15.4"/>
      </container>
   </containers>
```

Node Status and Interface Status Views

The Node Status and Interface Status Views are disabled by default. To use these views, the following must be true:

- Extended Topology must be enabled.
- APA must be enabled.
- The views must be enabled. See Enable or Disable the Status Views on page 117 for more information.

The views provide the ability to monitor node and interface status changes. Since status changes are key indicators of network faults, the Status Views can provide valuable information as you troubleshoot your network.

Operators can monitor status changes in real-time and manage the faulting nodes and interfaces through a lifecycle. The following functionality is available in the status views:

- Assign nodes and interfaces to users.
- Give each node and interface lifecycle state to provide an indication of where it is in the resolution lifecycle.
- Add notes to serve as a progress reminder or to communicate progress to other operators.

Before you can make user assignments, user authentication must be enabled. See the *Extended Topology and APA Deployment Guide* in the whitepaper directory for more information.

The Node Status or Interface Status Views provide different information, depending on how they are launched:

- When you launch the status views from Home Base, they show all nodes that have a non-normal status. These nodes are referred to as "Faulting Nodes" in the views.

- When you cross-launch the status views from another view, for example, a Neighbor View, the views show only the nodes or interfaces that were selected in the Neighbor View. These nodes are referred to as “Selected Nodes” in the views.

  ➤ The tab name in the view indicates if the view is populated by Faulting Nodes or Selected Nodes.

For more information on using the Node Status or Interface Status Views, see the online help.

Access Control

If user access is configured for containers, the access assignments apply to the Node Status and Interface Status Views. In these views, the nodes or interfaces a user can see are determined by the total set of nodes in all the containers to which the user has access. See the *Extended Topology and APA Deployment Guide* in the whitepaper directory for more information.

Audit Log File

When an operator makes one of the following changes to a status object, a user event is logged to an audit log file:

- Assignment made, changed, or unassigned
- Lifecycle state changed
- Notes added or changed

The audit log file is located as follows:

- *UNIX*:

```
/var/opt/OV/share/security/protected/logs/
NNM.DynamicViews.log
```

- *Windows*:

```
$OV_MAIN_PATH\share\security\protected\logs\
NNM.DynamicViews.log
```

The following format is present in the `NNM.DynamicViews.log` file:

- One row per log entry
- Row data is delineated with vertical bar (`|`) characters

- Each row contains the following information:
  - Sub-component name
  - Authenticated user name of current user
  - Client machine hostname
  - Status of this entry
  - Description of user action

Other audit log info is present in the `NNM.DynamicViews.log` file, but it has a different sub-component name.

Enable or Disable the Status Views

To enable the Status views, run the following script:

- *UNIX*:

  ```
  $OV_BIN/enableNSVandISV.ovpl
  ```

- *Windows*:

  ```
  %OV_BIN%\enableNSVandISV.ovpl
  ```

To disable the Status views, run the following script:

- *UNIX*:

  ```
  $OV_BIN/disableNSVandISV.ovpl
  ```

- *Windows*:

  ```
  %OV_BIN%\disableNSVandISV.ovpl
  ```

Alarm View

The Alarm View is added with the Extended Topology functionality when APA is enabled. The view allows you to monitor alarms so you can perform actions in the context of those alarms. The Alarm View provides a tabular list of alarms from the Alarm Server. By default, the alarms are ordered by time and show the same columns of information as the Java Alarm Browser.

This view is intended to be a replacement for the Java Alarm Browser. The Alarm View provides equivalent functionality to monitor alarms and includes many usability improvements. A difference is that the Alarm View does not show new alarms immediately. You can define an optional periodic refresh rate to determine how often new alarms are shown in the view, or you can choose to refresh the view manually. You can also select alarm categories from within the view, eliminating the need to launch a separate browser window for each selected category.

For more information on using the Alarm View, see the online help.

Automatic Refresh

A check box is provided to enable or disable auto-refresh of the Alarm View. You can specify the default interval by choosing the **Set Periodic Refresh Interval** option in the Options menu.

If you choose to use the auto refresh feature, keep in mind that the refresh option will likely scroll your alarm of focus from your immediate view when a refresh occurs. To keep your alarm of focus in the view, consider opening two Alarm Views - one with auto-refresh enabled and one with auto-refresh disabled. Your alarm of focus will stay in the view with auto-refresh disabled, and you will still see new alarms in the view with auto-refresh enabled.

Access Control

If user access is configured for containers, the access assignments apply to the Alarm View. In this view, the alarms a user can see are determined by the total set of nodes in all the containers to which the user has access. See the *Extended Topology and APA Deployment Guide* in the whitepaper directory for more information.

Neighbor View

The Neighbor View is available with or without Extended Topology enabled. This view provides a graphical representation of a selected device and the connector devices related to it, within a specified number of hops from the selected device.

If you have enabled Extended Topology, the Neighbor View shows enhanced layer 2 connectivity, more complete mouse-over information, and the ability to highlight VLANs.

For more information on using the Neighbor View, see the online help.

Node View

The Node View is available with or without Extended Topology enabled. This view allows you to define specific criteria to display only the nodes you want to see in the view.

If you have enabled Extended Topology, you can use new filters that are based on Extended Topology in the Node View. The default filter capability uses filters defined for classic NNM. The new filters replace these classic NNM filters.

You gain two key benefits when you use Extended Topology filters in Node View:

- Connectivity and filtering are consistent in the Node View. When you enable Extended Topology, the Node View shows Extended Topology connectivity and node information. When you enable Extended Topology filters, the filters are based on the same data as the Node View connectivity and node information. The result is a more consistent and accurate view of Extended Topology.
- Filter performance is improved. Extended Topology filters run much faster than classic NNM filters. For large topologies, they can radically reduce the typical filtering time of a Node View refresh.

When Extended Topology filters are enabled in the Node View, the list of filters in the Show Nodes selection list is replaced with the list of currently defined Extended Topology filters. Use the new filters the same as you used the classic NNM filters. For more information, see Use Extended Topology Filters in Node View on page 98.

You can define Extended Topology filters. See Chapter 5, Extended Topology Filters for more information.

For more information on using the Node View, see the online help.

Interface View

The Interface View is added with the Extended Topology functionality. This view helps you manage your network by creating exception views similar to Node View but with more detailed information.

For more information on using the Interface View, see the online help.

Path View

The Path View is available with or without Extended Topology enabled. This view provides details about the path between two nodes. The path is determined by calculating the active path that is in use at the time the view is requested.

If you have enabled Extended Topology, the Path View includes layer 2 devices in the calculated path. You can also tune the algorithms that compute the active path, using Extended Topology data. For more information, see Path Analysis with Path View on page 201.

For more information on using the Path View, see the online help.

VLAN View

The VLAN View is added with the Extended Topology functionality. This view presents a table that lists the VLANs and associated switches in your network. VLAN names can be meaningful or simple numeric identifiers.

For more information on VLAN View, see the online help.

Problem Diagnosis View

The Problem Diagnosis View is added with the Extended Topology functionality.

For more information on using the Problem Diagnosis View, see Problem Diagnosis is an automated IP network path analysis tool that presents end-to-end path information accurately. Furthermore, Problem Diagnosis lets you see detailed information from nodes and devices in a particular path. on page 204 or the online help.

HSRP View

The HSRP View is added with the Extended Topology functionality. To use this functionality, you must have a valid license for the Advanced Routing Smart Plug-in. The HSRP View presents a table of HSRP information. If you have configured Overlapping Address Domains, the top-level grouping is by OAD name.

For more information on using the HSRP View, see the online help.

VRRP View

The VRRP View is added with the Extended Topology functionality. This view presents a table of VRRP information. If you have configured Overlapping Address Domains, the top-level grouping is by OAD name.

For more information on using the VRRP View, see the online help.

Overlapping Address Domain View

The Overlapping Address Domain (OAD) View is added with the Extended Topology functionality. This view initially groups nodes within each OAD. You can expand each group to see the specific nodes that make up the group.

For more information on using the Overlapping Domain View, see the online help.

OSPF View

The OSPF View is added with the Extended Topology functionality. To use this functionality, you must have a valid license for the Advanced Routing Smart Plug-in.

If you have enabled the HP NNM/RAMS (Route Analytics Management System) Integration Module, this section does not apply to your situation.

You must manually configure your system to gather OSPF data. See Discovering OSPF Information on page 270 for more information.

The OSPF View offers both tabular and graphical representations of your OSPF areas. These views are useful when troubleshooting connectivity and performance problems in the network.

For more information on OSPF View, see the online help.

Modify Dynamic View Menus

Application developers and end-users can modify Dynamic View menus. See the *menusettings.xml (4)* reference page for more information. Details about how to access reference pages are in the online help.

Find a Web Browser on a UNIX System

If the Extended Topology installation program cannot find a web browser on a UNIX host system, you can specify the web browser location. This is the browser that launches the NNM web interface and the Extended Topology views. Edit the following file if you want to change the browser location after you install NNM and Extended Topology:

- *UNIX*: $OV_CONF/ovweb.conf

For more information about the ovweb.conf file, see the *ovweb.conf* reference page. Details about how to access reference pages are in the online help.

More Information on Dynamic Views

To review more detailed information on how to use Dynamic Views, refer to the online help.

4 Active Problem Analyzer

Basics of the Active Problem Analyzer

The Active Problem Analyzer (APA) monitors device status, analyzes problems in your network, and displays root cause information. To understand what APA monitors, you need to understand the following two cases:

- When Extended Topology is enabled but APA is not enabled to monitor your network, `netmon` is responsible to monitor every discovered device except for the following devices:
  - APA monitors Hot Standby Routing Protocol (HSRP) routers if you have a valid license for the Advanced Routing Smart Plug-in (SPI) for NNM to use the HSRP functionality. See Chapter 8, Advanced Routing Smart Plug-In for more information.
  - APA monitors all devices in your configured Overlapping Address Domains (OAD). See Chapter 9, Overlapping Address Domains for more information.
- When Extended Topology is enabled and APA is enabled to monitor your network, APA monitors every discovered device in your environment. See Enable or Disable APA on page 147 for more information.

When APA is not enabled to monitor your entire discovered network, APA still analyzes the connectivity details from the Extended Topology to provide you with accurate HSRP and OAD view status information. It provides you with this information in the following ways:

- Analyzes information from neighboring interfaces to provide better diagnostic information
- Generates alarms to indicate the root cause of an HSRP or OAD problem
- Maintains node, interface, and address status attributes
- Modifies Dynamic View node status colors

If you choose to have APA replace `netmon` and monitor your entire network, its default configuration allows it to poll devices as shown in Table 1. There are a number of factors to consider before you configure APA to replace `netmon`. This chapter will help you understand APA so you can make the correct decision for your environment.

Table 1 Default APA Polling Configuration

Device Type	SNMP Polling	ICMP Polling
Switch with No Connected Interfaces	No	Yes
Connected Interface on Router	Yes	Yes
Connected Interface on Switch	Yes	No
Connected Interface on End Node	No	Yes
Unconnected Interface on Router with Interface Administratively Up	Yes	Yes
Unconnected Interface on Router with Interface Administratively Down	No	No
Unconnected Interface on Switch with Interface Administratively Up	No	No
Unconnected Interface on Switch with Interface Administratively Down	No	No
Board	Yes	N/A

APA and the Netmon Process

If you prefer, you can continue to use the `netmon` process to monitor your network. Except for HSRP and OAD environments, which APA alone can monitor, classic NNM has always used `netmon` for this purpose. This is the default setting.

When you choose to use APA to monitor your network, `netmon` continues to run. If you have `netmon` configured to do so, netmon continues to do regular configuration polling and auto-discovery of new nodes. This means that both processes, `netmon` and APA, continue to run, but after APA is enabled, APA does the status polling.

It is important to remember that APA monitors with SNMP. For this reason, you do not want to monitor every device in your discovered environment or you will be overwhelmed with irrelevant alarms. For example, when people remove their laptops from the network at the end of the day, if those laptop connections are monitored, you would know about it with APA.

⚠ After APA is enabled, your `netmon` status polling configuration is no longer active. It is not transferable to APA. For example, if you gave an unmanaged status to specific interfaces when `netmon` was in charge of status polling, those interfaces do not automatically have an unmanaged status with APA. You must configure and tune APA in a different way. See the *Extended Topology and APA Deployment Guide* in the whitepaper directory for more information about tuning APA.

APA offers more accurate diagnostic information and faster throughput for greater scalability. The following list outlines the changes when you choose to have APA monitor your network:

- The `netmon` process no longer does status polling, but it still does device discovery.
- APA polls all of the IPv4 interfaces.
- As APA makes status changes to the Extended Topology, updates are sent for interfaces in the classic NNM topology. The `ovtopmd` process generates status events as it normally does.
- Dynamic Views get their information for each node from the Extended Topology if it is available. If Extended Topology does not have information, the `ovtopmd` process provides status information.

- If you have NNM installed as a management station with collection stations reporting to it, do not enable APA. You can enable APA on NNM only if it functions as a collection station or as a standalone management station.

Following is a list of benefits you receive when you use APA:

- APA analyzes information from neighboring interfaces and verifies that the failure is real before an alarm is generated.
- Link failures are identified based on Extended Topology connectivity.
- APA suppresses transient failures due to spanning tree reconvergence.
- Devices are polled with SNMP and ICMP when APA determines that it is appropriate for the situation.
- Polling performance is improved because APA ignores unconnected interfaces and utilizes multi-threaded processes.
- APA updates status information for objects in both `ovw` and Dynamic Views.

Table 2 compares APA with the `netmon` process. This should help clarify some of the differences between the `netmon` process and APA.

Table 2 Comparison of APA and the netmon Process

NNM with the netmon Process	NNM with APA Enabled
The polling list for NNM comes from `ovtopmd`.	The polling list for APA comes from the Extended Topology. Make sure your important nodes are not blocked by the `bridge.noDiscover` file.
NNM generates status alarms that are based on netmon's polling. There are no APA status alarms for devices in the general IP environment.	NNM generates status alarms from APA, which provide richer detail.
NNM alarms are correlated and appear in the Alarm Browser.	APA events participate in event correlation and appear in the Alarm Browser after verification.

Table 2 Comparison of APA and the netmon Process

NNM with the netmon Process	NNM with APA Enabled
Root cause analysis is based on netmon process information available to ECS. Alarms are sent for every device affected by a failure. Events that communicate topology changes to other NNM processes may be logged and not used by ECS.	Root cause analysis is based on APA information available to ECS. A single alarm is sent only for the actual root cause, not one for each affected device.
NNM does not model an IP address as a separate entity. The interface and IP address are modelled as a single interface object.	There are separate board, interface, and address objects. Extended Topology and APA allow for multiple boards per node, multiple interfaces per board, and multiple addresses per interface with each interface having its own status.
The `netmon` process derives device interface status with ICMP. You can add address information to the `netmon.snmpStatus` file to have NNM use SNMP status polling for specific interfaces. Doing this turns off ICMP polling for the targeted interfaces.	APA uses both ICMP and SNMP to determine interface status, depending on the APA configuration.
`ovw` shows propagated interface status. Status propagation is the sole basis for node, segment, and network status.	Some objects have their own basis for status. For example, HSRP analyzes the various status values to determine HSRP group status.
The `ovtopmd` process generates status change events for each device affected by a failure.	APA generates status alarms. It does not send secondary failure alarms. APA changes the status in the Extended Topology and only the root cause alarm is sent through the event system.

Table 2 Comparison of APA and the netmon Process

NNM with the netmon Process	NNM with APA Enabled
The `ipmap` process listens to the various status change alarms and updates node, interface, segment, and network status.	Dynamic Views listen for and display topology changes. APA sends Extended Topology node status to ovw nodes, segments, and networks in the classic NNM database.
Dynamic Views get status from the `ovtopmd` process for nodes and interfaces.	Dynamic Views get status from the Extended Topology. The `ovtopmd` process gets its IP status from APA.

Information Sharing Between Processes

The following information provides additional details on how APA and the `netmon` process cooperate with each other:

- APA can behave in two different ways:
  - It can monitor HSRP and OAD only, which is the default behavior.
  - You can enable APA to monitor everything that exists in the Extended Topology.
- If you enable APA, it forwards status information to both the `ovtopmd` process and Extended Topology. However, APA only forwards critical device status information to `ovtopmd`. As a result, NNM only displays node, interface, or address status and points to the possible root cause of the failure.
- Dynamic Views show Extended Topology status when that status is available; otherwise, the views show status that comes from the `ovtopmd` process. Devices that are not monitored by APA are given a `Not Monitored` status.
- A Neighbor View within an OAD shows APA status.

- If you view the Node Details or Interface Details pages of a device, it displays information as follows:
  - When APA monitors a node, the pages display Extended Topology and APA status information.
  - When the `netmon` process monitors a node, the page displays status from the `ovtopmd` process.

APA Failure Diagnosis

APA diagnoses network failures by doing additional failure analysis. During this failure analysis, APA does not generate multiple alarms for nodes that are not immediately adjacent to the detected fault. It attempts to generate one alarm that shows the root cause of a failure.

If APA determines that a node is down, it generates an OV_APA_NODE_DOWN alarm for the node and sets the status to critical. This alarm name implies a primary failure.

Failures detected on the other side of a known fault are designated as secondary failures and do not display at the top level in the Alarm Browser. You can see these secondary failures when you drill down from the corresponding primary failure.

Failure Analysis Details

During failure analysis, APA attempts to distinguish among the following three areas of the network:

- *Normal Area*: The area of the network near the management station where all the devices are operational and can be accessed via ICMP or SNMP.
- *Fault Area*: This area includes devices that contain a fault or are directly connected to a device downstream from the management station that contains a fault.

  This area will always contain at least one device that is up and responding to SNMP.

- *Far-From-Fault Area*: This area corresponds to devices that are downstream from the fault. That is, if you traverse a path from the management station to these devices, you will first pass through the Normal Area and the Fault Area and then arrive at the devices in the Far-From-Fault Area.

APA handles each of the three areas differently as follows:

- *Normal Area*: Devices located in this area are up and responding. Device statuses should be normal in the Extended Topology.

  You may see an OV_APA_<xxx>_UP alarm if a device was in the Fault Area and the fault has been fixed.

- *Fault Area*: APA analyzes this area, determines the most likely root cause object, and generates a single alarm for this object.

  For example, if APA determines that a node is down, it will emit an OV_APA_NODE_DOWN alarm for the node. In this example, APA considers objects contained by the node (interfaces, addresses, and boards) to be secondary failures and displays them with an unknown status in the Extended Topology. In this case, APA emits an OV_APA_<xxx>_UNREACHABLE alarm as appropriate.

  An interface in the Extended Topology may contain one or more addresses. If an interface status changes to down or unreachable, all contained addresses that are polled will change status to unreachable and an OV_APA_ADDR_UNREACHABLE alarm to be emitted.

  ➤ An alarm name that contains the term *DOWN*, as in OV_APA_NODE_DOWN, implies a primary failure.

  An alarm name containing the term *UNREACHABLE*, as in OV_APA_IF_UNREACHABLE, implies a secondary failure.

- *Far-From-Fault Area*: This area consists of devices that are not faulty but are affected by the fault. APA does not emit an alarm for these devices but sets the status to unknown in the Dynamic Views. APA inhibits alarm generation in this area to reduce clutter in the Alarm Browser.

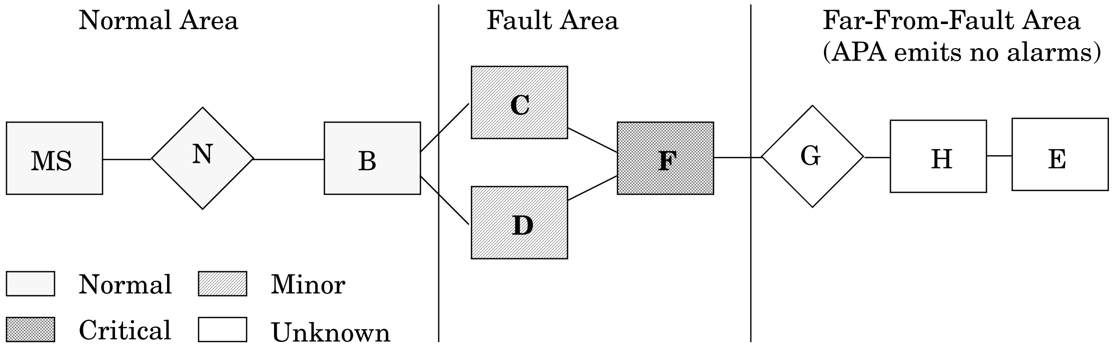
  ➤ Although APA excludes alarms from the Alarm Browser for all devices that are located in the Far-From-Fault area, you can change this behavior. You can configure APA to emit the alarms for important nodes that you designate.

To illustrate the performance improvement, suppose a fault occurs, resulting in 1,000 inaccessible nodes. Suppose each of these nodes contains four interfaces and five addresses. Although APA might only generate ten alarms for the Fault Area, it could emit ten alarms (one for the node, one for each interface, and one for each address) for each of the inaccessible nodes (in this case, 1000 nodes). In this example, without eliminating alarm generation, APA would have to change the status of 10,000 objects in the Far-From-Fault Area.

Refer to Figure 11 on page 135 for the following example:

Suppose that node F fails, and the management station (MS) cannot access nodes G, H, and E. The failure of node F causes all of Node F's interfaces to fail. This also causes the connected interfaces on nodes C and D to fail. The MS, node N, and node B remain up since they are accessible by the management station and are in the Normal Area.

Figure 11 APA Analysis Areas



When APA analyzes the Fault Area, it identifies node F as the root cause. APA considers the interfaces that connect nodes C and D to node F to be secondary failures. In this example, APA emits only one alarm, OV_APA_NODE_DOWN F, that is visible in the top level of the Alarm Browser. From the OV_APA_NODE_DOWN F alarm, you can drill down to the following alarms:

- OV_APA_CONNECTION_UNREACHABLE C.1 F.2
- OV_APA_CONNECTION_UNREACHABLE D.1 F.2

Note that the OV_APA_CONNECTION_UNREACHABLE alarms identify two interface objects as the target of the alarm. The OV_APA_NODE_DOWN and OV_APA_IF_DOWN alarms identify a single host and interface target object respectively.

In summary, APA finds the nodes and interfaces that are in the Fault Area, modifies the status appropriately, and sends out the appropriate alarms. In addition, APA finds the nodes in the Far-From-Fault Area and adjusts the status but does not generate any alarms.

Dynamic View Status

APA reports object status in the Dynamic Views using the colors shown in Table 3.

Table 3 APA Status Conditions

Dynamic View Status	Color	Dynamic View Status Description
Not Monitored	Tan	APA is not actively polling the object, such as a node, interface, connection, address, board, or aggregated port.
Unknown	Blue	APA cannot communicate with the object or considers the object to be a secondary failure. APA cannot determine the status of unreachable objects.
Disabled	Brown	The object's interface is administratively down.
Normal	Green	The object is functioning normally.
Minor	Yellow	The object is functioning but an other contained object, such as an interface, connection, address, board, or aggregated port object is not functioning.
Critical	Red	The object is not functioning normally.

SNMP Traps from Network Devices

You can configure the SNMP agents on many network devices to send SNMP traps to an NNM management station's hostname or IP address. In many cases, when APA receives traps, it initiates an immediate poll of the device generating the SNMP trap. When APA receives any of the following traps, it will take the associated action:

- `SNMP_Link_Down` or `SNMP_Link_Up`

  Action: APA analyzes the source's interfaces.

- `SNMP_Cold_Start` or `SNMP_Warm_Start`

  Action: APA analyzes the source and its configuration.

- `Cisco_Link_Down` or `Cisco_Link_Up`

  Action: APA analyzes the source's interfaces.

- `Cisco_Cold_Start` or `Cisco_Warm_Start`

  Action: APA analyzes the source's interfaces and its configuration.

- `Cisco_HsrpStateChange`

  Action: APA analyzes the HSRP group.

- `Cisco_ModuleUp` or `Cisco_ModuleDown`

  Action: APA analyzes the source and its configuration.

- `ciscoLS1010ChassisChangeNotification`

  Action: APA analyzes the source and its configuration.

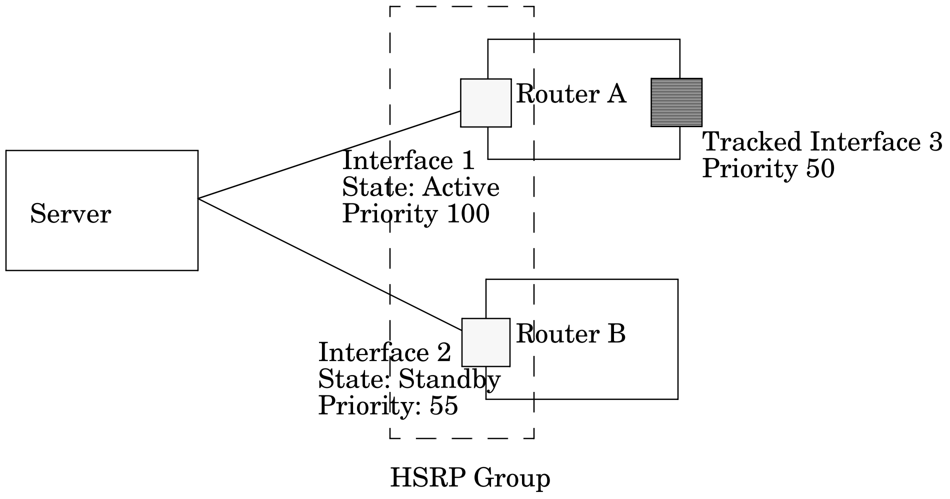
APA and View Status

The information contained in this section discusses how APA affects the displayed status in various Dynamic Views.

APA and HSRP Status

APA polls HSRP routers and reports HSRP group status information. APA also monitors tracked interfaces as configured on the routers contained in the HSRP group. For example, in Figure 12, Router A contains an active interface 1 in the HSRP group with a priority of 100. Router B is a standby router in the HSRP group and contains interface 2 with a priority of 55. If the tracked interface 3 fails, interface 1's priority falls to 50 (100 minus 50), and interface 2 becomes the active interface. The HSRP view displays the new priority value and the interface state changes.

Figure 12 How APA Monitors Tracked Interfaces



APA, Layer 3 Polling, and Node Status

APA only updates the status of a device or interface that is identified as the primary failure. APA also only synchronizes the primary node or interface status with the `ovtopmd` process and the `ovw` user interface. The result is that APA only updates the `ovw` user interface with the primary fault status. After a faulty device is functioning properly, APA polls the device, and the status returns to normal.

APA considers nodes located outside of the Fault Area to be secondary failures. APA does not update the `ovw` user interface with the status of these nodes.

APA does not support IPX networked devices, IPX interfaces, or any devices not residing in the hosts.nnm file. APA does support OAD devices.

Board Status and the ovw User Interface

APA polls the board status of a node with SNMP. Suppose that APA reports a board down, and there are no interfaces down within the board or within the node itself. In this example, NNM Advanced Edition sets the Extended Topology node status to minor and generates an OV_APA_BOARD_DOWN alarm. However, the node status in the ovw user interface remains normal, as the ovw user interface does not take board status into consideration when determining node status.

APA and Event Reduction

When APA is enabled and properly configured, it reduces the number of alarms you see in your Alarm Browser through the use of APA correlators, which are accessed from the Correlation Composer user interface. The information provided in this section is from the context of the operator mode of the HP OpenView Correlation Composer user interface. For more information about HP OpenView Correlation Composer, see *HP OpenView Correlation Composer's Guide*.

In the Correlation Composer, NNM includes a set of built-in correlators that are enabled out-of-the-box. These basic correlators are described in detail in the Event Reduction chapter of *Managing Your Network*.

This section explains additional NNM correlators that are enabled when APA is configured.

Syslog Integration and APA Correlators

Syslog Integration and APA are distinct pieces of functionality provided with NNM Advanced Edition. Each can be enabled individually; however, the most benefit is achieved when Syslog Integration is enabled in conjunction with APA functionality.

Many of the correlators provided with APA work on syslog events. For example, certain syslog messages cause an immediate APA poll of the network device identified in the message. Other correlators listen for syslog messages and nest them under the appropriate APA event, enabling better root cause analysis.

If Syslog Integration functionality is not enabled but APA is enabled, some of the APA correlators may not be used.

APA Alarms

When you enable APA, it disables the status polling feature of the `netmon` process and establishes the `ovet_poll` process as the polling engine. The `ovet_poll` process contains alarm types beginning with OV_APA. The following list describes the alarms that APA may display or log:

- *OV_APA_ADDR_DOWN*: APA generates this alarm when it detects that a network entity's address status goes from *up* to *down*.
- *OV_APA_ADDR_Intermittent*: APA generates this alarm when it detects that a network address's status has gone *down* and *up* multiple times.
- *OV_APA_ADDR_UNREACHABLE*: APA generates this alarm when it detects that an address is unreachable due to another failure, such as the address's interface being down.
- *OV_APA_ADDR_UP*: APA generates this alarm when it detects that a network entity's address status goes from *down* to *up*.
- *OV_APA_AGGPORTCONN_DOWN*: APA generates this alarm when the aggregate port connection between two nodes is not responding to polls and all interfaces may be down on both sides of the connection.
- *OV_APA_AGGPORTCONN_UNREACHABLE*: APA generates this alarm when the aggregate port connection between two nodes is not responding to polls on both sides of the connection. The problem is probably due to another entity.
- *OV_APA_AGGPORTCONN_UP*: APA generates this alarm when the aggregate port connection between two nodes is responding to polls and no interfaces are down on either side of the connection.
- *OV_APA_AGGPORT_DEGRADED*: APA generates this alarm when the aggregate port connection between two nodes is responding to polls and some of the interfaces are down.
- *OV_APA_AGGPORT_DISABLED*: APA generates this alarm when the aggregate port is not responding to polls in a normal fashion. This could be because all the interfaces' `ifAdminStatus` are *down* or *testing*. This aggregate port is the primary failure.
- *OV_APA_AGGPORT_DOWN:* APA generates this alarm when the aggregate port connection between two nodes is not responding to polls and all interfaces may be down.
- *OV_APA_AGGPORT_UNREACHABLE*: APA generates this alarm when the aggregate port connection between two nodes is not responding to polls. The problem is probably due to another entity.
- *OV_APA_AGGPORT_UP*: APA generates this alarm when the aggregate port connection between two nodes is responding to polls and no interfaces are down.

- *OV_APA_AGGPORT_NOTDEGRADED*: APA generates this alarm when the aggregate port connection between two nodes is responding to polls and all of the interfaces are up.
- *OV_APA_AGGPORT_REMOVED*: APA generates this alarm when the aggregate port detects that an SNMP noSuchObj error is returned from monitoring queries of this aggregated port network interface.
- *OV_APA_BOARD_DOWN*: APA generates this alarm when it detects that a node's board status goes from *up* to *down*.
- *OV_APA_BOARD_REMOVED*: APA generates this alarm when it detects that a node's board has been removed.
- *OV_APA_BOARD_UNREACHABLE*: APA generates this alarm when it detects that a node's board stops responding to polls due to another entity in the network.
- *OV_APA_BOARD_UP*: generates this alarm when it detects that a node's board status goes from *down* to *up*
- *OV_APA_CONNECTION_DOWN*: APA generates this alarm when it detects that a connection's status goes from *up* to *down*.
- *OV_APA_CONNECTION_Intermittent*: APA generates this alarm when it detects that a network connection's status has gone *down* and *up* multiple times.
- *OV_APA_CONNECTION_UNREACHABLE*: APA generates this alarm when it detects that a connection is unreachable due to another failure.
- *OV_APA_CONNECTION_UP*: APA generates this alarm when it detects that a connection's status goes from *down* to *up*.
- *OV_APA_DEMAND_POLL*: HP OpenView processes generate this alarm when they want APA to poll a specific network entity.
- *OV_APA_IF_DISABLED*: This specific alarm indicates that the interface is not responding to polls in a normal fashion. This could be because the interface's `ifAdminStatus` is *down* or *testing*.
- *OV_APA_IF_DOWN*: APA generates this alarm when it detects that a network entity's interface status goes from *up* to *down*.
- *OV_APA_IF_Intermittent*: APA generates this alarm when it detects that an interface's status has gone *down* and *up* multiple times.
- *OV_APA_IF_UNREACHABLE*: APA generates this alarm when it detects that an interface is unreachable due to another failure.

- *OV_APA_IF_UP*: APA generates this alarm when it detects that a network entity's interface status goes from *down* to *up*.
- *OV_APA_IF_REMOVED*: APA generates this alarm when it detects that an SNMP noSuchObj error is returned from monitoring queries of this network interface.
- *OV_APA_Message*: APA generates this alarm to reflect changes in the network that are best communicated through text. This alarm is mainly used for internal troubleshooting.
- *OV_APA_NODE_DOWN*: APA generates this alarm when it detects that a node's status goes from *up* to *down*.
- *OV_APA_NODE_Intermittent*: APA generates this alarm when it detects that a node's status has gone *down* and *up* multiple times.
- *OV_APA_NODE_RENUMBERING*: APA generates this alarm when it detects that the interfaces or boards on a node were renumbered.
- *OV_APA_NODE_RENUMBERING_FIXED*: APA generates this alarm when it detects that the interfaces or boards on a node no longer appear to have been renumbered. This can happen after a rediscovery, and will clear the prior *OV_APA_NODE_RENUMBERING* alarm for this node.
- *OV_APA_NODE_UNREACHABLE*: APA generates this alarm when it detects that a node is unreachable due to another failure, such as and upstream node being down.
- *OV_APA_NODE_UP*: APA generates this alarm when it detects that a node's status goes from *down* to *up*.
- *OV_APA_NODE_SNMP_NOT_RESPONDING*: APA generates this alarm when it detects that node is not responding to APA SNMP query.
- *OV_APA_POLL*: HP OpenView processes generate this alarm when they want APA to poll a specific network entity.
- *OV_APA_Statistics*: APA generates this alarm to report various execution statistics that are useful for monitoring its performance.
- *OV_APA_ISLAND_GROUP_DOWN*: APA identifies an island group as a group of connected nodes that may not be connected to the management station in Extended Topology. APA generates this alarm when it detects that all the nodes in an island group are DOWN.

- *OV_APA_ISLAND_GROUP_UP*: APA identifies an island group as a group of connected nodes that may not be connected to the management station in Extended Topology. APA generates this alarm when it detects that at least one node in an island group is UP. This happens only after an *OV_APA_ISLAND_GROUP_DOWN* event is generated for that particular group.
- *OV_APA_PE_THRESH*: APA generates this event when it detects that the Polling Engine queue size has breached a threshold value.
- *OV_APA_PE_REARM*: APA generates this event when it detects that the Polling Engine queue size has dropped below a configured level. This happens only after an event occurs because a value was breached.
- *OV_APA_SA_THRESH*: APA generates this event when it detects that the Status Analyzer queue size has breached a threshold value.
- *OV_APA_SA_REARM*: APA generates this event when it detects that the Status Analyzer queue size has dropped below a configured level. This happens only after an event occurs because a value was breached.

APA does not generate the IntermittentStatus alarms. They are generated by the OV_PollerIntermittent correlators. This only happens when both of the following occur:

1. The OV_PollerIntermittent correlators are enabled.
2. APA generates multiple down events within the time window specified in those correlators.

APA Correlators

APA-enabled correlators are logically divided into seven namespaces. Following is an overview of each namespace:

- OV_PollerIntermittent Namespace

  The OV_PollerIntermittent namespace contains a set of correlators that notify you when a network component is flapping. When APA status change events or link down/link up transitions exceed a count threshold during a specified time interval, flapping events are generated. One flapping event is generated for each type of object: link, node, interface, address, or connection.

- OV_PollerLinkDown Namespace

  The OV_PollerLinkDown namespace contains a set of correlators that determine if the root cause of a link down event is a result of a node or connection that is down. When appropriate, link down events are correlated under the APA root cause alarm and then removed from the Alarm Browser.

- OV_CiscoBoard Namespace

  The OV_CiscoBoard namespace contains a set of correlators that determine if link down events, syslog link down messages, and line protocol down messages are secondary events to board down or module down events. Secondary events are correlated under the root cause alarm, identifying the board or module that failed.

- OV_PollerBoardDown Namespace

  The OV_PollerBoardDown namespace contains a set of correlators that determine if board down events or syslog board down messages should be nested under the root cause APA board down alarm. When appropriate, the board down alarms are correlated under the APA board down alarm and then removed from the Alarm Browser.

- OV_PollerTrigger Namespace

  The OV_PollerTrigger namespace contains a set of correlators that trigger an immediate APA poll and analysis of a network entity - node, interface, board, or HSRP group. When certain types of traps or syslog messages arrive, these correlators generate the appropriate APA poll trigger request on the network entity. Depending on the type of event, the correlators may issue a status poll, configuration poll, or force poll request.

- OV_PollerTriggerCorr Namespace

  The OV_PollerTriggerCorr namespace contains a set of correlators that correlate duplicate poll trigger events not already being correlated by the other APA correlators. Secondary events are correlated under the first event to arrive or under the APA alarm that is generated as a result of the poll request.

- OV_PollerPortAgg Namespace

  The OV_PollerPortAgg namespace contains a set of correlators that determine if link down, syslog down, and syslog port aggregation events are secondary events to APA port aggregation status alarms. Secondary events are correlated under the APA root cause alarm.

For information on correlators you can use with APA, view the online help for the Correlation Composer or review the *HP Correlation Composer's Guide*.

Enable or Disable APA

When you choose to enable APA to monitor your entire discovered IPv4 network, the polling features of the `netmon` process are disabled. When the polling features of the `netmon` process are disabled, the polling configuration you set up for `netmon` is not transferable to APA and is no longer active.

Before you enable APA, it is a good idea to complete a few initial configuration tasks for APA. If you turn on APA without any tuning, APA will generate a large number of alarms until the tuning is done. See the *Extended Topology and APA Deployment Guide* in the whitepaper directory for more information on basic APA tuning.

APA takes its polling list exclusively from the Extended Topology, which includes all the IPv4 interfaces that are discovered by the `netmon` process and placed in the `hosts.nnm` file.

Enable APA

To enable APA polling and disable polling using the `netmon` process, as Administrator or root, run the following script:

- *UNIX*:

```
$OV_BIN/ovet_apaConfig.ovpl -enable APAPolling
```

- *Windows*:

```
%OV_BIN%\ovet_apaConfig.ovpl -enable APAPolling
```

See the *ovet_apaConfig.ovpl* reference page for more information. Details about how to access reference pages are in the online help.

Disable APA

To disable APA and enable polling using the `netmon` process, as Administrator or root, run the following script:

- *UNIX*:

```
$OV_BIN/ovet_apaConfig.ovpl -disable APAPolling
```

- *Windows*:

  ```
  %OV_BIN%\ovet_apaConfig.ovpl -disable APAPolling
  ```

➤ To learn if APA is enabled, run the following command:

```
ovet_apaConfig.ovpl -query APAPolling
```

See the *ovet_apaConfig.ovpl* reference page for more information. Details about how to access reference pages are in the online help.

➤ When you disable APA status polling, APA continues to monitor addresses that belong to an OAD, query HSRP routers, and report HSRP group status information.

Interface Configuration Change Detection

Some device configuration activities may cause SNMP agents to renumber SNMP MIB interface instance values of some Ethernet switches and routers. These activities include the following actions:

- Moving a board from one slot to another
- Removing a board
- Adding a new board
- Removing the supervisor board in a dual supervisor board system
- Power cycling a switch or router

APA collects and stores MIB information from Ethernet switches and routers. This includes information from MIB instances such as ifAlias, ifName, ifDescr, and ifPhysAddress. APA checks to see if this indexed information has changed since the last configuration check. Any deviation from the last ifAlias, ifName, ifDescr, or ifPhysAddress values can be a symptom caused by a device's interfaces being renumbered.

If APA determines an interface change has occurred, it generates the following alarm:

- OV_APA_NODE_RENUMBERING

APA also checks for added interfaces, removed interfaces, and removed boards during status polls. The following alarms are generated when one of these activities occurs:

- OV_APA_IF_ADDED
- OV_APA_IF_REMOVED
- OV_APA_BOARD_REMOVED

Any of the interface configuration change alarms listed in this section can be the triggers that cause the `ovet_bridge` process to do an attribute update.

APA logs the details about each alarm in the following file:

- *UNIX*:

  ```
  $OV_PRIV_LOG/ovet_poll.log.txt
  ```

- *Windows*:

  ```
  %OV_PRIV_LOG%\data\ovet_poll.log.txt
  ```

➤ You can update the status of a device and check to see if interface renumbering activity has occurred by using the following procedure:

1. Select a device from one of the Dynamic Views.
2. Right-click the mouse and select **APA Status Poll**.

Enable Interface Configuration Change Detection

To enable APA to automatically monitor devices for interface configuration changes, complete the following steps:

1. Make a backup copy of the following file:
   - *UNIX*:

     ```
     $OV_CONF/nnmet/paConfig.xml
     ```

   - *Windows*:

     ```
     %OV_CONF%\nnmet\paConfig.xml
     ```

2. Open the `paConfig.xml` file for editing.
3. By default, APA does not query the `entLastChangeTime` parameter during status polls. To enable this feature, complete the following steps:
   a. Find the `entityMibEnable` parameter in the `ConfigPollSettings` section in `paConfig.xml` file:

   ```
   <parameter>
      <name>entityMibEnable</name>
      <title>
         Trigger config poll by checking entityMib changes.
      </title>
      <description>
         Enable/Disable query of entityMib during status
         polls for a device. If Enabled, config poll is
         triggered when the value of entityMib changes.
      </description>
      <varValue>
         <varType>Bool</varType>
         <value>true</value>
      </varValue>
   ```

```
</parameter>
```

b Set the value for this parameter to `true` to enable APA to query the `entLastChangeTime` during status polls.

> If you choose to have APA query the `entLastChangeTime` parameter during status polls, we recommend that you create a filter to identify only those devices that support the entLastChangeTime MIB OID. Then you can enable this parameter only for those devices specified by the filter. For more information on polling policies and filters, see the *Extended Topology and APA Deployment Guide* in the whitepaper directory.

4 By default, APA does not perform configuration polls for end nodes. To enable this feature, complete the following steps:

a Find the `isEndNode` filter in `ConfigPollSettings` section of the `paConfig.xml` file:

```
<classSpecification>
   <filterName>isEndNode</filterName>
   <parameterList>
      <parameter>
         <name>enable</name>
         <title>Enable config poll for device</title>
         <description>
            Enable/Disable config poll of a device
         </description>
         <varValue>
            <varType>Bool</varType>
            <value>true</value>
         </varValue>
      </parameter>
   </parameterList>
</classSpecification>
```

b Set the value for this parameter to `true` to enable APA to perform configuration polls for end nodes.

➤ If you choose to have APA perform configuration polls for end nodes, we recommend that you add the `interfaceDetailFields` parameter in the `isEndNode` filter and query only the required fields during configuration polls on end nodes. If you query non-configured MIBS for endnodes, the result is a `No-Such_Obj` error that is returned by the SNMP agent of the endNode. In this case, APA sends renumber events if the `NoSuchObjectEnabled` flag has a value of true.

Following is an example of the `interfaceDetailFields` parameter:

```
<parameter>
   <name>interfaceDetailFields</name>
   <title>Interface detail fields to check</title>
   <description>
      Fields to use in interface renumbering check.
      Possible values are:
      ifAlias,
      ifName,
      ifPhysAddress,
      ifDescr
      One or more of these can be specified in this
      parameter, separated by a comma.
   </description>
   <varValue>
      <varType>String</varType>
      <value>ifDescr</value>
   </varValue>
</parameter>
```

5 By default, APA does not query the ifNumber parameter during regular status polls. If enabled, APA sends an alarm when the value of the ifNumber changes. To enable this feature, complete the following steps:

a. Find the `ifNumberQueryEnable` parameter in the `ConfigPollSettings` section in `paConfig.xml` file:

```
<parameter>
   <name>ifNumberQueryEnable</name>
   <title>Enable ifNumber query</title>
   <description>
      Enable/Disable query of Interface Numbers. If
      enabled, ifNumber is also queried during regular
      status polls of APA. Any change in the ifNumber will
      trigger APA to send an event (OV_APA_IF_ADDED).
   </description>
   <varValue>
      <varType>Bool</varType>
      <value>true</value>
   </varValue>
</parameter>
```

b. Set the value for this parameter to `true` to enable APA to query the ifNumber parameter during status polls.

APA Performance

You can monitor the performance of APA from Home Base. If the performance of APA is not acceptable, you can configure APA to perform better in your environment.

The most important enhancement to performance is gained when you initially configure APA. There are basic configuration concepts you should apply before you turn on APA. If you do not tune APA before you enable it, you may be flooded with alarms until you take the basic configuration steps. See the *Extended Topology and APA Deployment Guide* in the whitepaper directory for more information.

Beyond the basic configuration steps outlined in the *Extended Topology and APA Deployment Guide*, there are other things you can do to enhance the performance of APA. Those actions are covered in this section.

Monitor APA Performance From Home Base

NNM collects information that shows how APA is performing. From Home Base, select the `Polling/Summary Analysis` tab to see information from the Active Problem Analyzer. APA statistics are collected on five-minute intervals. The statistics are as follows:

- *Active Analyzer Tasks*: This represents the number of polling results that are currently under analysis. This number should trend toward zero when the network is stable. If this number never trends toward zero, you may need to increase the number of threads in the status analyzer thread pool.

  If the number of `Active Analyzer Tasks` begins to steadily increase, you are experiencing network problems. Look in the Alarm Browser for an alarm that indicates the root cause of your network problem.

- *Waiting Poller Tasks*: This represents the maximum number of polling tasks that were waiting to be completed during the last polling interval. Track the quantity of `Waiting Poller Tasks` that is normal for your environment. If this number begins to increase, it indicates that the APA poller is unable to keep up with the polling load.

  To remedy this problem, review the following list and make adjustments if necessary.

  - Make sure NNM and Extended Topology are only monitoring your critical devices.
  - Increase the default polling interval.
  - Increase the polling intervals of any of the device classes.
  - Increase the polling engine thread pool size.

- *Addresses Polled (ICMP)*: This represents the number of addresses that were pinged during the last statistics reporting interval. Track the quantity of `Addresses Polled (ICMP)` that is normal for your environment. This number is an indication of how busy your APA polling engine is.
- *Interfaces Polled (SNMP)*: This represents the number of interfaces that were queried for status through SNMP during the last reporting interval. Track the number of `Interfaces Polled (SNMP)` that is normal for your environment. This number is an indicator of how busy your APA polling engine is.
- *Waiting Analyzer Tasks*: This represents the number of polling results waiting to be analyzed. When a series of failures occur, this number rises. If this number continues to rise over several polling cycles, it indicates a serious issue in the network. A temporary surge that trends toward zero is normal when a fault or change occurs.
- *HSRP Groups Polled*: This represents the number of HSRP groups that were queried for status during the last reporting interval. Track the `HSRP Groups Polled` number that is normal for your environment. This number is an indication of how busy your APA polling engine is.
- *Boards Polled*: This represents the number of boards that were queried for status through SNMP during the last reporting interval. Track the number of Boards Polled (SNMP) that is normal for your environment.This number is an indicator of how busy your APA polling engine is.

Allow or Suppress APA Status Polling

You can configure NNM to allow or suppress the APA status polling of objects with the `ovet_toposet` command. You can also allow or suppress filtering from any dynamic view. See *Allow or Suppress Monitoring* in the online help for more information.

Table 4 shows the affect on objects when you allow or suppress APA status polling for nodes, boards, interfaces, or addresses.

Table 4 Expected Object Status Behavior when Suppressing or Allowing APA Status Polling

Allowed or Suppressed APA Polling of Object	APA Polling Behavior
Node	Allows or suppresses APA status polling for the boards associated with a node, all of the interfaces associated with each board, and the addresses associated with each interface.
Board	Allows or suppresses APA status polling for the interfaces associated with a board and the addresses associated with each interface.
Interface	Allows or suppresses APA status polling for the addresses associated with an interface.
Address	Allows or suppresses APA status polling for the specified address.

➤ If you use the `ovet_toposet` command or user interface to allow an object to be polled, there are other parameters within the `paConfig.xml` file that could override the `ovet_toposet` command and suppress the polling of the targeted object.

See the *ovet_toposet* reference page for more information about the ovet_toposet command. Details about how to access reference pages are in the online help. You can also allow or suppress filtering from any dynamic view. See *Allow or Suppress Monitoring* in the online help for more information.

➤ APA propagates the status of objects considered to be the root cause of a fault to `ovw`.

APA may propagate the status of unpolled interfaces, objects not considered to be the root cause of a fault, or objects that do not exist in the Extended Topology as having a normal or unknown status in `ovw`.

APA Queue Threshold Event

This feature helps you monitor the status analyzer and polling engine queue size and generates an event in case either queue size breaches a specified threshold value. These events can be used to take appropriate action when APA goes into an unresponsive state due to a very large queue size.

By configuring the appropriate threshold value, we can monitor the state of APA and take appropriate action in such a scenario.

The events in Table 5 are generated for the Status Analyzer and Polling Engine queues.

Table 5 Events for Status Analyzer and Polling Engine Queries

Events	State
OV_APA_SA_THRESH	Status Analyzer queue size has exceeded the upper threshold value
OV_APA_SA_REARM	Status Analyzer queue size is lower than the threshold value
OV_APA_PE_THRESH	Polling Engine queue size has exceeded the upper threshold value
OV_APA_PE_ REARM	Polling Engine queue size is lower than the threshold value

The parameters in Table 6 are added to specify the upper and lower threshold limits for the Status Analyzer and Polling Engine queues.

Table 6 Parameters that Specify Threshold Limits for Status Analyzer and Polling Engine Queries

Parameter	Description
saQueueSelfMonitoringThreshold	Upper threshold value for Status Analyzer queue

Table 6 Parameters that Specify Threshold Limits for Status Analyzer and Polling Engine Queries

Parameter	Description
saQueueSelfMonitoringRearm	Lower threshold value for Status Analyzer queue
peQueueSelfMonitoringThreshold	Upper threshold value for Polling Engine queue
peQueueSelfMonitoringRearm	Lower threshold value for Polling Engine queue

Change Default Value

As administrator or root, complete the following steps to change the default value for the peQueueSelfMonitoringThreshold.

To change the value for the other three parameters, use these same steps with the thread pool text for those parameters.

1 As Administrator or root, edit the `paConfig.xml` file. See Location of the APA Configuration File on page 168 to learn the location of the file.

Before you edit the `paConfig.xml` file, it is a good idea to make a backup copy in case something goes wrong. Even a small syntax error in this file could cause APA to fail.

2 Look for the following polling engine thread pool text:

```
<parameter>
   <name>peQueueSelfMonitoringThreshold</name>
   <title>Polling Engine Queue Threshold</title>
   <description>
      . . .
   </description>
   <varValue>
      <varType>Integer</varType>
      <value>30000</value>
   </varValue>
</parameter>
```

3 Change the bold number value to the threshold value you want and save your changes.

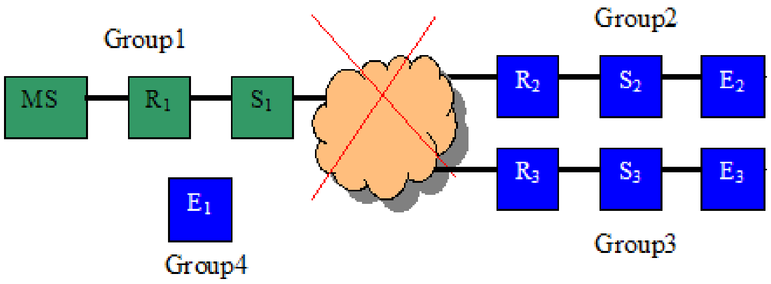
4 Stop and restart the `ovet_poll` process to apply your changes. See Stop and Restart Processes on page 17 for more information.

APA Island Group Monitoring

APA tracks groups of connected devices and reports when a group goes down with an OV_APA_ISLAND_GROUP_DOWN alarm. Alternately, the OV_APA_ISLAND_GROUP_UP alarm occurs when the group is up again. In the case of these alarms, an island is defined as a group of connected devices that NNM displays in a group that is not connected to the rest of the topology.

An example of an environment with multiple islands is a financial institution with many branches. Each branch is connected to other branches with a WAN connection, but the WAN connection is not discovered by Extended Topology so it appears that each branch is an isolated island of nodes in the NNM topology.

The following example illustrates the concept of an island group. There are four groups that are not connected. Group 4 contains only a single island node.



Extended Topology discovers each device and the connectivity, but it does not discover the groups. APA observes connectivity as it loads devices from Extended Topology, and it builds a set of groups within APA. With the groups in place, APA can issue alarms for the groups.

When all the nodes in a group go down (red or blue), APA issues a group down alarm. When at least one node in the group is back up (green or yellow), APA issues a group up alarm. These alarms are integrated into the ECS Pairwise circuit.

When an island group alarm is generated, a node is selected to represent the group. This group representative is typically a router, is automatically selected by APA, and is the object referenced in the alarm label.

The OV_APA_ISLAND_GROUP_DOWN alarm is not generated for groups that contain only one node. In this case, we call the node an island node.

Island Group Configuration

APA assigns two configurable parameters for every group:

1 A representative node for the group.

2 An empty label that you can change to be a descriptive name for the group.

For information on how to configure island groups, see the E*xtended Topology and APA Deployment Guide* in the whitepaper directory.

Enable/Disable Island Group Monitoring

For information on how to enable or disable island group monitoring, see the E*xtended Topology and APA Deployment Guide* in the whitepaper directory,

Virtual Routing Redundancy Protocol and APA Polling

By default, APA polls VRRP groups. VRRP devices are monitored in a way that is similar to HSRP devices, and APA generates the same state change events for both HSRP and VRRP devices. Specific trap values of HSRP events are reused for VRRP support.

Ten protocol specific var-binds are added to the existing HSRP events. These var-binds can be used to determine if events are generated from an HSRP group or VRRP group.

The display format for the existing HSRP events is changed from “HSRP router group” to “RRP router group”.

Events

The following enterprises are added for VRRP Protocol support:

- vrrpMIB (.1.3.6.1.2.1.68)
- rcVrrpNotifications (.1.3.6.1.2.1.46.1.3)
- foundry (.1.3.6.1.4.1.1991)

The following events are added for VRRP protocol support:

- EVENT rcVrrpTrapNewMaster
- EVENT rcVrrpTrapAuthFailure
- EVENT rcVrrpTrapStateTransition
- EVENT vrrpTrapNewMaster
- EVENT vrrpTrapAuthFailure
- EVENT snTrapVrrpIfStateChange

The following existing APA HSRP events are enhanced for VRRP Protocol support:

- EVENT OV_HSRP_No_Active
- EVENT OV_HSRP_Multiple_Active
- EVENT OV_HSRP_NoStandby
- EVENT OV_HSRP_Degraded

- EVENT OV_HSRP_Standby_Changed
- EVENT OV_HSRP_Normal
- EVENT OV_HSRP_Multiple_Standby

The following ten new var-binds have been added to the existing HSRP events to support VRRP protocol. These var-binds start from sequence 20.

20 Redundant Router Protocol (for example, VRRP)

21 Redundant Router Protocol Version (for example, Nortel Rapid City)

22 Protocol specific state name for generic state Initial if applicable

23 Protocol specific state name for generic state Learn if applicable

24 Protocol specific state name for generic state Listen if applicable

25 Protocol specific state name for generic state Speak if applicable

26 Protocol specific state name for generic state Standby if applicable

27 Protocol specific state name for generic state Active if applicable

28 Protocol specific state name for generic state (future) if applicable

29 Protocol specific state name for generic state (future) if applicable

Values of these var-binds contain protocol specific values. For example, if an event is generated for a Nortel VRRP group, the 20th var-bind value will be "VRRP", the 21st var-bind value will be "Nortel", the 22nd var-bind value will be "Initialize", and the 23rd var-bind value will be an empty string. The other var-binds will be populated similarly.

ECS

Two factstores are a part of VRRP support:

- XrpCorrelate
- XrpTriggerPoll

The XrpCorrelate Correlator store has the following Correlators:

1 OV_VRRP_APA_Corr - Enterprise : 1.3.6.1.4.11.2.17.1 Specific_trap :

    60001409 (OV_HSRP_No_Active)

    60001410 (OV_HSRP_Multiple_Active)

60001411 (OV_HSRP_No_Standby)

60001412 (OV_HSRP_Degraded)

60001413 (OV_HSRP_FailOver)

60001414 (OV_HSRP_Standby_Changed)

60001415 (OV_HSRP_Normal)

60001416 (OV_HSRP_Multiple_Standby)

2 OV_Foundry_VRRP_Events - Enterprise : 1.3.6.1.4.1.1991 Specific_trap :

    34 (snTrapVrrpIfStateChange)

3 OV_Juniper_VRRP_Events - Enterprise : 1.3.6.1.2.1.68 Specific_trap :

    1 (vrrpTrapNewMaster)

4 OV_RC_VRRP_Events - Enterprise : 1.3.6.1.2.1.46.1.3 Specific_trap :

    1 (rcVrrpTrapNewMaster)

    2 (rcVrrpTrapAuthFailure)

    3 (rcVrrpTrapStateTransition)

The Poller Trigger Correlations provide correlation of the trigger events. Trigger events are the events that trigger an APA poll, and they are correlated with any APA status events that are generated as a result of the trigger event. The APA status event is displayed as the top-level event in the Alarm Browser, and the trigger events are correlated under it.

The XrpTriggerPoll Correlator store has the following Correlators:

1 OV_Juniper_VRRP_Poll - Enterprise : 1.3.6.1.2.1.68 Specific_trap :

    1 (vrrpTrapNewMaster)

2 OV_Foundry_VRRP_Poll - Enterprise : 1.3.6.1.4.1.1991 Specific_trap :

    34 (snTrapVrrpIfStateChange)

3 OV_RC_VRRP_Poll - Enterprise : 1.3.6.1.2.1.46.1.3 Specific_trap :

    1 (rcVrrpTrapNewMaster)

    2 (rcVrrpTrapAuthFailure)

    3 (rcVrrpTrapStateTransition)

When certain traps or syslog messages are received from a network device, an APA poll and analysis should be scheduled immediately rather than waiting until the next poll cycle. APA is configured to receive certain traps directly, but there are other traps and syslog messages that require further processing before a poll is requested. The PollerTrigger correlators perform this additional processing and generate the appropriate poll trigger request.

Poll trigger requests can be one of the following four types:

1. Poll if network entity up – a down indication has been received as a trap or syslog message, and this request is for the poller to poll the device if the current APA status is up.
2. Poll if network entity down – an up indication has been received as a trap or syslog message, and this request is for the poller to poll the device if the current APA status is down.
3. Poll regardless of the current APA status.
4. Perform a configuration poll, for example, if a node is rebooted.

There are three types of network entities that can be specified in the poll trigger request: node, interface, and card. This tells APA which type of network entity should be polled.

These PollerTrigger correlators also keep track of whether or not the same poll request has been sent previously to APA during the period defined for each correlator. This prevents multiple identical requests from being sent to APA during the period.

Aggregation and APA Polling

Aggregation occurs when redundant connections are bundled and treated like a single connection. This configuration increases the bandwidth and reliability between two nodes. APA supports two aggregate port MIBs, PagP and MLT. APA behavior is similar for each kind of aggregation, but there are a few differences.

Extended Topology discovers all of the physical interfaces that connect two nodes and then creates a new interface to represent the group. This new interface is called the logical interface. When a physical interface fails, packets can still be sent between the two nodes because of the redundancy. Loss of a connection reduces bandwidth, but the network continues to function.

The differences between PagP and MLT include the following:

- Logical interface placement in the ifTable
    - PagP - The logical interface is in the ifTable. This means that when the ifOperStatus of an interface is down, the logical interface turns Critical/Red.
    - MLT - The MLT logical interface does not exist in the ifTable so interfaces do not turn yellow or red.
- For MLT, there will never be an OV_APA_IF_DOWN alarm for a logical interface.

APA polls the physical interfaces that participate in aggregation, but APA does not poll the aggregate logical interfaces. Since the aggregate logical interface is not polled, the status displays as Not Monitored. The standard color for objects that are not polled is Eggshell.

When a participating physical interface fails but the node and board do not fail, the following happens:

- The status of the aggregate logical interface remains Not Monitored/ Eggshell.
- The status of the physical interface turns to Critical/Red.
- The status of the monitored containers (node and board) turn to Minor/ Yellow, indicating a failure within the container.

When a participating physical interface fails, the following alarms may be generated:

- OV_APA_AGGPORT_DEGRADED
- *OV_APA_IF_DOWN

  The '*' indicates that the alarm is correlated and only visible when you drill down in an OV_APA_AGGPORT_DEGRADED alarm.

When a physical interface failure is resolved, the alarms clear from the xnmevents alarm browser.

When a container fails (board or node), an OV_APA_AGGPORT_DEGRADED alarm will not be generated. In this case, a root cause alarm may be generated, and the interface alarm will be correlated under it. For example, if the node fails, the following alarms may be generated:

- OV_APA_NODE_DOWN
- *OV_APA_IF_UNREACHABLE

  The '*' indicates that the alarm is correlated and only visible when you drill down in an OV_APA_NODE_DOWN alarm

The node will turn Critical/Red to indicate it is the primary failure. The interface will turn Unknown/Blue to indicate it is the secondary failure. APA uses the status Unknown and the term Unreachable to indicate secondary failure.

Location of the APA Configuration File

When you modify the APA configuration file, it is important to backup the original APA configuration file in case your modifications cause problems. Even a small syntax error in this file could cause APA to fail.

You can find the APA configuration file at the following location:

- UNIX:

  ```
  $OV_CONF/nnmet/paConfig.xml
  ```

- Windows:

  ```
  %OV_CONF%\nnmet\paConfig.xml
  ```

Frequently Asked Questions

When enabled, APA replaces several features normally provided by the `netmon` process. Below is a list of frequently asked questions and answers that describe APA features.

How does APA affect NNM performance?

Answer: APA is multi-threaded, suppresses secondary alarms, and generates one alarm for a failure's root cause in most cases. This results in a faster, more scalable polling engine.

How does APA derive node and interface status?

Answer: APA separates the polling of IP addresses and interfaces. It uses ICMP to poll addresses and SNMP to poll interfaces. Their status is derived from the results of these polls. For interfaces that have one or more IP addresses associated with them, the status of the interface reflects the status of the IP addresses as well. For example, if the interface status is normal but one or more of the associated IP addresses do not respond, the interface status is set to minor. For an interface that is not monitored, if all the associated addresses do not respond, the interface status is set to critical, but if all the addresses respond. the interface status is set to normal.

What does APA do in an HSRP environment?

Answer: Extended Topology with the Advanced Routing SPI queries the devices that participate in an HSRP group. These devices must support the HSRP protocol. APA retrieves specific HSRP MIB values that provide the actual HSRP roles and responsibilities of each device in the group. Using this information, APA generates HSRP alarms when HSRP status changes occur.

Does APA consider Spanning Tree Protocol (STP) when analyzing failures?

Answer: APA considers the effects of the STP when it diagnoses the root cause of unresponsive interfaces. It suppresses transient failures due to spanning tree reconvergence which results in fewer transient alarms. APA waits a short period of time before analysis is performed to allow the STP to converge and bring up any backup links.

Does APA consider meshed switches when diagnosing network faults?

Answer: APA analyzes meshed environments when calculating the root cause of unresponsive interfaces.

Does APA use any of the layer 2 information from the netmon process?

Answer: No. APA uses the layer 2 connectivity information stored in the Extended Topology database. It is important to remember that APA still relies on the `netmon` process for device discovery. The Extended Topology discovery process provides connectivity.

How do you configure SNMP information for APA?

Answer: APA uses the same SNMP configuration information as the `netmon` process.

Can APA ping the virtual IP address?

Answer: This is not supported in this release.

How does APA interact with incremental discovery and zones?

Answer: APA becomes aware of any of the topology changes Extended Topology discovers. For example, after Extended Topology initiates and completes a discovery or you tell Extended Topology to discover a specific zone and it completes that discovery, APA immediately uses the new information. However, APA itself is not aware of zones.

5 Extended Topology Filters

Basics of Extended Topology Filters

Extended Topology discovers a variety of devices that are connected to your network infrastructure, including routers, switches, and computers. It also discovers interfaces and cards that are a part of these devices. When you use Extended Topology filters, you can interact with defined subsets of your network.

Proper use of topology filters can enhance your experience with Dynamic Views, including Node view and Container view. You can also use topology filters to configure APA polling options.

An Extended Topology filter is comprised of assertions and/or other filters that are combined by logical operators. The purpose of a filter is to create a defined subset of objects from the devices discovered by Extended Topology. A single filter can only group together objects of the same type.

From a high level, you can specify a group of devices with a named filter. You can specify the filter on a command line as an argument or from the user interface. XML is the underlying implementation technology. For basic use, XML knowledge is not critical. However, if you plan to modify or enhance the basic filters, a strong understanding of XML is required.

Extended Topology Database

At the core of Extended Topology is a set of internal APIs and programs, some of which are exposed. A major component of the current implementation of Extended Topology is a Relational Database Management System (RDBMS) that holds much of the topology information.

You can access the Extended Topology database schema at the following location:

- *UNIX*:

```
$OV_CONF/analysis/sqlScripts/tables_TopoService.schema
```

- *Windows*:

```
%OV_CONF%\analysis\sqlScripts\tables_TopoService.schema
```

This schema is the basic framework for the database. Columns and tables are added when you enable Extended Topology. NNM SPIs can also extend this schema. See documentation for NNM SPIs for more information.

Numerous processes and files are involved in the management of the data. There are complex interdependencies; therefore, direct access to the RDBMS is not supported. Use the tools provided to access the information. No other access method is supported or recommended.

Filter Types

There are many ways to select an appropriate subset of your devices, including the following.

- Products from a Particular Vendor
- DNS Node Names
- IP Addresses and Node Names
- Extensible Attributes
- System Object IDs
- VLAN Port Type
- Reachability State
- Administrative or Operational State
- MIB Type
- Address Mode

You can further refine your filters by combining several of the attributes listed above.

Assertions and Filters

The basic building blocks of Extended Topology filters are called assertions. Assertions are similar to filters and contain selection criteria that are directly tied to data elements that are stored in the NNM database.

The following assertions are available to help you define your filters:

- nodeAssertion
  - Use this assertion for node objects.
- interfaceAssertion
  - Use this assertion for interface objects.
- cardAssertion
  - Use this assertion for card/board objects.
- ifcAssertion
  - Use this assertion for interface container objects.
- hsrpgroupAssertion
  - Use this assertion for HSRP group objects.
- addressAssertion
  - Use this assertion for address objects.

Filters may contain combinations of assertions or other filters. This allows for more complex query capabilities.

➤ Only filters on the same type of object can be combined by logical operators. To combine objects of different types, use an association filter. See Filter with Association Types on page 189 for more information.

In filters, the objectType attribute is used to determine if nodes or interfaces should be returned. When filters and assertions are contained within subfilters, the appropriate object type is automatically returned.

Create or Modify a Filter or Assertion

Filters and assertions are very similar. Assertions are a kind of filter. When you review the Structure of TopoFilters.xml on page 177, notice that both filters and assertions are included in the `<filters>` tag.

To create or modify a filter or assertion, complete the following steps:

1. Make a backup of the following file:
   - *UNIX*:

     ```
     $OV_CONF/nnmet/topology/filter/TopoFilters.xml
     ```

   - *Windows*:

     ```
     %OV_CONF%\nnmet\topology\filter\TopoFilters.xml
     ```

2. Open the `TopoFilters.xml` file for editing.
3. If you want to create a new filter or assertion, find a one that is similar to the new filter or assertion that you want to create, and copy that filter or assertion. Paste the copy in the `TopoFilters.xml` file.

   When you decide to modify a filter, carefully consider the impact of the change. Do not modify filters that are defined by HP software. These filters can have an impact on APA polling configurations.

   - Filter Placement in `TopoFilters.xml` - Filters must be placed together above the assertions section. It does not matter where you place a new filter in the list of filters because objects can match multiple filters.
   - Assertion Placement in `TopoFilters.xml` - Assertions must be placed with other assertions in the assertions section. Place assertions of the same type together in the section.
4. Modify the filter or assertion to create the filter or assertion you want.
5. Save your changes and close the file.

If you want more information about the way filters are defined, you can review the XSD schema definition files, `TopoFilter.xsd` and `TopoFilterCommonTypeDef.xsd`. Both files are located in the same directory as `TopoFilters.xml`.

Verify a Filter or Assertion

After you create or modify a filter or assertion, it is a good idea to test it. To test your filter or assertion, complete the following steps:

1 Make sure your new filter or assertion shows up in the list of available filters with the following command:

```
ovet_topodump.ovpl -lfilt
```

If you have an error in your syntax, this command fails and provides diagnostic information to help you find the error.

2 After your filter or assertion shows up in the list of available filters, use the `ovet_topodump.ovpl` script to view a list of objects that pass the filter. For example, the following usage displays a list of nodes that pass a filter:

```
ovet_topodump.ovpl -node -filt <filter_name>
```

See the *ovet_topodump.ovpl* reference page for more information. Details about how to access reference pages are in the online help.

XML Examples

Structure of TopoFilters.xml

The top-level structure of the `TopoFilters.xml` file is as follows:

```
<filters...>
   <!-- The set of filter definitions. -->
   <filter name="name"...>
   </filter>
   ...
   <!-- The set of assertion definitions. -->
   <nodeAssertion name="name"...>
   </nodeAssertion>
   <interfaceAssertion name="name"...>
   </interfaceAssertion>
   <addressAssertion name="name"...>
   </addressAssertion>
   ...
</filters>
```

Node Assertions

This section lists attributes you can use in Node Assertions and provides an example of how to use Node Assertions.

Node Assertion Attributes

Table 7 provides examples of XML tags used in the supplied filters for Node Assertions.

Table 7 Node Assertion Attributes

Tag	Allowable Data	Description
<name>	string	• valid DNS name • may contain '*' wildcards
<SysName>	string	• system name of the node

Table 7 Node Assertion Attributes

Tag	Allowable Data	Description
<lastUpdateTimeUTC>	integer	• last time the node was updated • time_t format
<description>	string	• description of the node
<IPAddress>	IPv4 IPv6	• IP addresses on the node • v4 ranges and v6 "::" are OK • OADId is also allowed
<sysOID>	dotted integer string	• system object ID of the node • may contain '*' wildcard at end of pattern
<capability>	"isSNMPSupported" "isMPLS" "isHSRP" "isLanSwitch" "isIPv4Router" "isIPv6Router" "isMultiHome" "isATM" "isFrameRelay" "isSTP" "isVRRP" "isWireless" "isRMON" "isRMON2" "isDS1" "isDS3" "isSONET" "isCDP" "isBGP" "isL2Connected" "isOSPF" "isNewNode"	• capability of the node

Table 7 Node Assertion Attributes

Tag	Allowable Data	Description
<status>	"normal" "warning" "critical" "disabled" "minor" "major" "unknown" "marginal"	• overall status of the node • use with equal, atLeast, or upTo operator
<extensibleAttribute>	Examples: MPLSId - for nodeAssertion	• extensible attributes of the node
<HostIDFile>	URI	• predefined host name or IP address in the file • must be in special XML format

Node Assertion Example

```
<nodeAssertion name="isHSRP" description="Hot Standby Router
Protocol devices">
   <operator oper="NOOP">
      <attribute>
         <capability>isHSRP</capability>
      </attribute>
   </operator>
</nodeAssertion>
```

Interface Assertions

This section lists attributes you can use in Interface Assertions and provides an example of how to use Interface Assertions.

Interface Assertion Attributes

Table 8 provides examples of XML tags used in the supplied filters for Interface Assertions.

Table 8 Interface Assertion Attributes

Tag	Allowable Data	Description
<lastUpdateTimeUTC>	integer	• last time the interface was updated • time_t format
<IPAddress>	IPv4 IPv6	• IP address bound to the interface • v4 ranges and v6 "::" are OK • OADId is also allowed
<ifDescription>	string	• description of the interface
<ifAlias>	string	• interface name alias
<ifName>	string	• interface name
<ifIndex>	integer	• interface index
<ifDesc>	string	• description of the interface
<vlanPortType>	multipleVlanPort noVlanPort oneVlanPort	• role of this port in VLAN config - trunk, access, or no VLAN
<ifAdminState>	"up" "down" "testing"	• interface administration state
<ifOperStatus>	"up" "down" "testing" "unknown" "dormant" "notPresent" "lowerLayerDown"	• interface operational status
<ifType>	integer	• interface type

Table 8 Interface Assertion Attributes

Tag	Allowable Data	Description
<ifSpeed>	integer	• interface speed
<status>	"normal" "warning" "critical" "disabled" "minor" "major" "unknown" "marginal"	• overall status of the interface • use with equal, atLeast, or upTo operator
<extensibleAttribute>	Examples: MPLSName - for interfaceAssertion	• extensible attributes of the interface
<capability>	"isMPLS" "isHSRP" "isATM" "isFrameRelay" "isSTP" "isVRRP" "isWireless" "isDS1" "isDS3" "isSONET" "isCDP" "isBGP" "isAccessPort" "isNewInterface" "isL2Connected" "isOSPF" "isMeshInterface" "isMultiCast" "isAggregatedIF" "isPartOfAggregatedIF" "isTunnel"	• capability of the interface

Interface Assertion Example

```
<interfaceAssertion name="MyIFName" title="IFNAME"
description="Name of the interface">
   <operator oper="NOOP">
      <attribute>
         <ifName>Fa0/1</ifName>
      </attribute>
   </operator>
</interfaceAssertion>
```

Card Assertions

This section lists attributes you can use in Card Assertions and provides an example of how to use Card Assertions.

Card Assertion Attributes

Table 9 provides examples of XML tags used in the supplied filters for Card Assertions.

Table 9 Card Assertion Attributes

Tag	Allowable Data	Description
<name>	string	• NNM entity name of the card
<lastUpdateTimeUTC>	integer	• last time the node was updated • time_t format
<index>	integer	• card index
<description>	string	• description of the card
<type>	string	• MIB type field on the card
<model>	string	• MIB model field on the card
<sn>	string	• MIB serial number field on the card
<fwversion>	string	• MIB firmware version field on the card

Table 9 Card Assertion Attributes

Tag	Allowable Data	Description
<hwversion>	string	• MIB hardware version field on the card
<swversion>	string	• MIB software version field on the card
<componentName>	string	• match on SNMP system object ID of the card
<status>	"normal" "warning" "critical" "disabled" "minor" "major" "unknown" "marginal"	• overall status of the card • use with equal, atLeast, or upTo operator
<extensibleAttribute>		• extensible attributes of the card
<cardAdminStatus>	"unset" "up" "down" "unknown" "other" "no_such_object" "not_responding" "not_available"	• card administration status
<cardOperStatus>	"unset" "up" "down" "minor_fault" "testing" "other" "unknown" "not_present" "no_such_object"	• card operational status

Table 9 Card Assertion Attributes

Tag	Allowable Data	Description
<mibType>	cisco_stack_mib cisco_rhino_mib cisco_c2900_mib unknown_mib	• MIB type from which the card data was read

Card Assertion Example

```
<cardAssertion name="STACKMIB" title="CISCO STACK MIB"
description="Board with MIB type CISCO STACK">
   <operator oper="NOOP">
      <attribute>
         <mibType>cisco_stack_mib</mibType>
      </attribute>
   </operator>
</cardAssertion>
```

Interface Container Assertions

This section lists attributes you can use in Interface Container Assertions and provides an example of how to use Interface Container Assertions.

An interface container represents a logical or virtual grouping of physical interfaces. An interface container might be used to model VPNs and VLANs.

Interface Container Assertion Attributes

Table 10 provides examples of XML tags used in the supplied filters for Interface Container Assertions.

Table 10 Interface Container Assertion Attributes

Tag	Allowable Data	Description
<name>	string	• NNM entity name of the interface container

Table 10 Interface Container Assertion Attributes

Tag	Allowable Data	Description
<description>	string	• description of the interface container
<lastUpdateTimeUTC>	integer	• last time the interface container was updated • time_t format
<type>	integer	• type of the interface container
<status>	"normal" "warning" "critical" "disabled" "minor" "major" "unknown" "marginal"	• overall status of the interface container • use with equal, atLeast, or upTo operator
<extensibleAttribute>		• extensible attributes of the interface container

Interface Container Assertion Example

```
<ifcAssertion name="MyIFCAssertion" description="test filter
for IFC assertion" title="">
   <operator oper="NOOP">
      <attribute>
         <name>IFCAssertionTest</name>
      </attribute>
   </operator>
</ifcAssertion>
```

HSRP Group Assertions

This section lists attributes you can use in HSRP Group Assertions and provides an example of how to use HSRP Group Assertions.

HSRP Group Assertion Attributes

Table 11 provides examples of XML tags used in the supplied filters for HSRP Group Assertions.

Table 11 HSRP Group Assertion Attributes

Tag	Allowable Data	Description
<virtualAddress>	IPv4 IPv6	• virtual address of the HSRP group

HSRP Group Assertion Example

```
<hsrpgrpAssertion name="MyHSRPgrpAssertion" description="test
filter for HSRPgrp" title="">
   <operator oper="NOOP">
      <attribute>
         <virtualAddress>
            <IPv4>
               <address>10.0.0.1</address>
            </IPv4>
         </virtualAddress>
      </attribute>
   </operator>
</hsrpgrpAssertion>
```

Address Assertions

This section lists attributes you can use in Address Assertions and provides an example of how to use Address Assertions.

Address Assertion Attributes

Table 12 provides examples of XML tags used in the supplied filters for Address Assertions.

Table 12 Address Assertion Attributes

Tag	Allowable Data	Description
<IPAddress>	IPv4 IPv6	• IP address • v4 ranges and v6 "::" are OK • OADId is also allowed
<reachabilityState>	"responding" "notResponding" "unreachable" "agentDown" "disabled"	• state of the address
<extensibleAttribute>		• extensible attributes of the address

Address Assertion Example

```
<addressAssertion name="AllVirtualAddress"
title="AllVirtualAddress" description="AllVirtualAddress">
   <operator oper="NOOP">
      <attribute>
         <IPAddress>
            <IPv4>
               <capability>isVirtualAddress</capability>
            </IPv4>
         </IPAddress>
      </attribute>
   </operator>
</addressAssertion>
```

Extended Topology Filter Examples

Filter Operators

➤ Only filters on the same type of object can be combined by logical operators. To combine objects of different types, use an association filter. See Filter with Association Types on page 189 for more information.

Valid operators are “OR”, “AND”, “NOT” & “NOOP”

```
<filter name="InfrDev" objectType="Node"
title="InfrastructureDevice" description="Infrastructure
devices including both switches and router">
   <operator oper="OR">
      <filterName>isSwitch</filterName>
      <filterName>isRouter</filterName>
   </operator>
</filter>

<filter name="UnconnectedAdminUpRouterIF"
objectType="Interface"
title="UnconnectedAdminUpRouterInterface"
description="Unconnected Interface in a router which has an
IF Admin Status of up">
   <operator oper="AND">
      <filterName>InterfaceInRouter</filterName>
      <filterName>NotConnectedIF</filterName>
      <filterName>AdminUpInterface</filterName>
   </operator>
</filter>

<nodeAssertion name="isEndNode" title="isEndNode"
description="End Nodes (Non-Routers or Switches)">
   <operator oper="NOT">
      <operator oper="OR">
         <attribute>
            <capability>isIPv4Router</capability>
         </attribute>
         <attribute>
            <capability>isIPv6Router</capability>
         </attribute>
```

```
            <attribute>
               <capability>isLanSwitch</capability>
            </attribute>
         </operator>
      </operator>
   </nodeAssertion>
```

Filter with Association Types

The following example uses the association type, “contains,” to combine a node and an interface assertion.

```
<nodeAssertion name="NodeWithDownInterface"
title="NodeWithDownInterface" description="Nodes with at
least one Interface in Admin Down State">
   <operator oper="NOOP">
      <nodeAssociation ascType="contains">
         AdminDownInterface
      </nodeAssociation>
   </operator>
</nodeAssertion>
<interfaceAssertion name="AdminDownInterface"
title="AdminDownInterface" description="Interface with
administrative state down">
   <operator oper="NOOP">
      <attribute>
         <ifAdminState>
            down
         </ifAdminState>
      </attribute>
   </operator>
</interfaceAssertion>
```

Filter Products From a Particular Vendor

The following sample node assertion filters products from a specified vender.

```
<nodeAssertion name="CiscoDevices" title="CiscoDevices"
description="All Cisco devices">
   <operator oper="NOOP">
      <attribute>
         <sysOID>1.3.6.1.4.1.9.*</sysOID>
      </attribute>
   </operator>
</nodeAssertion>
```

Filter by IP Addresses and Node Names with HostIDFile

The following sample node assertion brings together the important nodes in your network. The nodes are identified by IP address and name in the HostIDFile designated as MyHostID.xml.

```
<nodeAssertion name="myPetNodes" title="myPetNodes"
description="Nodes I'm interested in">
   <operator oper="NOOP">
      <attribute>
         <HostIDFile>MyHostID.xml</HostIDFile>
      </attribute>
   </operator>
</nodeAssertion>
```

Sample HostIDFile

```
<?xml version="1.0" encoding="UTF-8"?>
<HostIDs xmlns="http://www.hp.com/openview/NetworkTopology/
TopologyFilter" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation= "http://www.hp.com/
openview/NetworkTopology/TopologyFilterHostIDFile.xsd">
   <DNSName>*cisco*</DNSName>
   <DNSName>hsrp*.cnd.hp.com</DNSName>
   <IPv4>
      <address>10.0.0.*</address>
      <OADId>33</OADId>
   </IPv4
   <IPv4>
      <address>15.2.135.62-100</address>
   </IPv4>
</HostIDs>
```

Filter by IP Address Range with an Address Assertion

The following sample filter and addressAssertion can be used together to create a filter containing all nodes of a particular IP Address range. This example finds all nodes having IP addresses that match the pattern 10.*.*.*:

```
<filter name="SampleAddressFilter" objectType="Node"
title="Sample address filter" description="All nodes with IP
Address 10.*.*.*">
   <operator oper="NOOP">
      <filterName>SampleAddressAssertion</filterName>
   </operator>
</filter>
<addressAssertion name="SampleAddressAssertion" title="Sample
address assertion" description="All 10.*.*.* addresses">
   <operator oper="NOOP">
      <attribute>
         <IPAddress>
            <IPv4><address>10.*.*.*</address></IPv4>
         </IPAddress>
      </attribute>
   </operator>
</addressAssertion>
```

Filter by IP Address Range with a Node Assertion

The following sample nodeAssertion can be used to create a filter that selects all nodes that have an address in a range or a wildcard name. Note that this assertion selects a node if it has an address in the defined range. This is not the management address or the DNS address, and the name is the actual name in the database, not necessarily the DNS name.

```
<nodeAssertion name="group1" title="group1"
description="Group of nodes based on name or IP address">
   <operator oper="OR">
      <attribute>
         <name>
            <string>*dgr*</string>
         </name>
      </attribute>
      <attribute>
         <IPAddress>
            <IPv4>
               <address>10.102.*.*</address>
            </IPv4>
         </IPAddress>
      </attribute>
   </operator>
</nodeAssertion>
```

Filter by Node Name with a Regular Expression

The following sample nodeAssertion can be used to create a filter containing all nodes with a name that matches a specified regular expression. This example finds all nodes whose names contain the string “ci”:

```
<nodeAssertion name="SampleNodeNameFilter" title="Sample node
name filter" description="All node names containing the
string 'ci'">
   <operator oper="NOOP">
      <attribute>
         <name><string>*ci*</string></name>
      </attribute>
   </operator>
</nodeAssertion>
```

Filter by Attribute

The following sample nodeAssertion can be used to create a filter containing all nodes that match a SysName and a name. Sysname is a value from the MIB. The name value is the name for the node in the Extended Topology database. This name could have come from another source such as DNS.

```
<nodeAssertion name="FortCollins" title="FortCollins"
description="Fort Collins">
   <operator oper="OR">
      <attribute>
         <SysName>abcon*</SysName>
      </attribute>
      <attribute>
         <name>
            <string>ns.corp.com</string>
         </name>
      </attribute>
      <attribute>
         <SysName>*sshsrv*</SysName>
      </attribute>
   </operator>
</nodeAssertion>
```

Filter by sysOID and IPAddress

The following sample nodeAssertion can be used to create a filter for nodes that have an address in a range and have a certain sysOID. The logic is ((sysOID | sysOID | sysOID) && (addrRange | addrRange)).

```
<nodeAssertion name="ColoradoServers" title="ColoradoServers"
description="Colorado Servers">
   <operator oper="AND">
      <operator oper="OR">
         <attribute>
            <sysOID>1.3.6.1.4.1.311.*</sysOID>
         </attribute>
         <attribute>
            <sysOID>1.3.6.1.4.1.11.*</sysOID>
         </attribute>
         <attribute>
            <sysOID>1.3.6.1.4.1.42.*</sysOID>
         </attribute>
      </operator> <!-- END "OR" -->
      <operator oper="OR">
         <attribute>
            <IPAddress>
               <IPv4>
                  <address>10.102.*.*</address>
               </IPv4>
            </IPAddress>
         </attribute>
         <attribute>
            <IPAddress>
               <IPv4>
                  <address>10.32.121-122.*</address>
               </IPv4>
            </IPAddress>
         </attribute>
      </operator> <!-- END OR -->
   </operator> <!-- END AND -->
</nodeAssertion>
```

Filter by Interfaces on a Filtered Node

The following sample assertions can be used to create a filter that returns an associated interface on a filtered node. This filter can be used to set a polling policy on nodes of a certain type and, subsequently, a polling policy on the interfaces on those nodes. The nodeAssertion identifies the nodes with the SysName attribute. Next, the interfaceAssertion identifies interfaces on these nodes.

```
<nodeAssertion name="WiFi-AGR" title="WiFi-AGR"
description="Aggregator for Wi-Fi">
   <operator oper="OR">
      <attribute>
         <SysName>al*</SysName>
      </attribute>
      <attribute>
         <SysName>bo*</SysName>
      </attribute>
   </operator>
</nodeAssertion>
<interfaceAssertion name="IFWiFi-AGR" title="IFWiFi-AGR"
description="Interfaces for Wi-Fi">
   <operator oper="NOOP">
      <interfaceAssociation ascType="inNode">
         WiFi-AGR
      </interfaceAssociation>
   </operator>
</interfaceAssertion>
```

Filter by Interface Name

The following sample interfaceAssertion can be used to create a filter that returns interfaces that have a name and are not a loopback.

```
<interfaceAssertion name="IFNameNotLo0" description="IF Not
Loopback0">
   <operator oper="NOT">
      <attribute><ifName>Lo0</ifName></attribute>
   </operator>
</interfaceAssertion>
```

Filter by Letters in a Name

The following sample interfaceAssertion can be used to create a filter that returns each interface with a name that has “Gi” in it.

```
<interfaceAssertion name="IFgig" description="IF Not
Loopback0">
   <operator oper="NOOP">
      <attribute>
         <ifName>*Gi*</ifName>
      </attribute>
   </operator>
</interfaceAssertion>
```

Filter by Interface Description

The following sample interfaceAssertion can be used to create a filter that returns interfaces with defined descriptions.

```
<interfaceAssertion name="IFDescVlan" description="IF
Description filter">
   <operator oper="OR">
      <attribute>
         <ifDescription>*unrouted*</ifDescription>
      </attribute>
      <attribute>
         <ifDescription>*virtual*</ifDescription>
      </attribute>
   </operator>
</interfaceAssertion>
```

Best Practices

Should I use a filter or an assertion?

Filters and assertions are very similar. Assertions are a kind of filter. When you review the Structure of TopoFilters.xml on page 177, notice that both filters and assertions are included in the `<filters>` tag.

Use a filter when you want to use multiple assertion filters as one filter. The following example illustrates this concept:

```
<filter name="CiscoRouter" objectType="Node"
title="CiscoRouter" description="Cisco Router">
   <operator oper="AND">
      <filterName>isRouter</filterName>
      <filterName>CiscoDevices</filterName>
   </operator>
</filter>
```

In this example, the output of the CiscoRouter filter is the sum of the output obtained from the two filters, isRouter and CiscoDevices, which are nothing more than node assertions. The CiscoRouter filter depends on two properly defined node assertions.

When the CiscoRouter filter is processed, it happens in steps. The CiscoRouter filter is resolved first, and then the two assertion filters are resolved to further define the filter. In general, you can think of an assertion as the most basic piece of a filter.

Which assertion should I use?

Select the type of assertion in light of your requirements. For example, to filter node objects, use a node assertion; to filter address objects, use an address assertion; to filter interface objects, use an interface assertion; and so on.

Following are examples of good practice with assertions:

- To filter interfaces with duplicate IP addresses, use the predefined interface assertion, InterfacesWithDupAddrs, which is built with the isDuplicated capability.
- To filter nodes that are either IPv4 or IPv6 routers, use the predefined node assertion, isRouter, which is built with isIPv4Router and isIPv6Router capabilities.

What is allowed in a filter or assertion?

`TopoFilter.xsd` controls the syntax and defines the set of rules for filters and assertions. A filter that has correct syntax is not necessarily guaranteed to produce output. A filter also depends on translators (*.xsl) and source code logic.

Following are examples of restrictions that `TopoFilter.xsd` puts on filters:

- By default, the number of assertion attributes cannot exceed 32. This value can be modified, but use extreme caution if you modify this file.
- The format for IP addresses is defined as follows: 129.83.64.*, 64.128.2.71-72, and so on.
- For each object type, a subset of available attributes applies. For example, OperStatus and AdminState only apply to interface objects.

Frequently Asked Questions

When enabled, Extended Topology can provide a wealth of information about elements in your network. Below are frequently asked questions and answers about accessing this information with the filtering mechanism.

Is there a tool that makes working with XML easier?

Answer: Many tools are available to work with XML files. Some of them support validation against the .xsd file. In addition some editors support XML markup. Use of any of these tools will facilitate changes to the filter file.

How efficient are filters?

Answer: The filters are converted into queries in the Topology data store. In order to provide a flexible and powerful syntax, some of the resulting queries may not be very efficient. In these cases, we have traded efficiency for power.

Can I create my own filter file?

Answer: No. Several production components use the existing filters that ship with the product. In addition, the programs that interpret these filters require a fixed location for TopoFilters.xml (and other supporting files). You can add to the existing filters. However, do not alter any of the existing entries since this could cause unexpected failures.

Can I use my NNM filters from earlier NNM versions?

Answer: Not directly. Extended Topology filters have been engineered to provide a more powerful and extensible mechanism for filtering. However, you should be able to create an equivalent filter using the new syntax in most cases.

Why is querying the RDBMS directly not allowed?

Answer: It is likely that the tables and schemas used internally by Extended Topology will be changed in new releases of NNM. Only the supported tools should be used to access Extended Topology information. The internal NNM data stores are proprietary information and are protected by copyright and patents. Direct use of this data is strictly prohibited.

Can I modify the schema definition file (.xsd) or XML transformation (.xsl) file?

Answer: Do not modify either of these files. If you incorrectly modify either file, you can cause NNM to fail.

6 Path Analysis

Path Analysis with Path View

Path View shows the paths utilized in your network. This section explains how you can enable algorithms that cause the Path View to use information from Extended Topology to compute paths. When you choose this option, your paths are more accurate and robust. In addition, improvements are also made to enhance logging and debugging.

Path View is based on SNMP queries. If you want a device to participate in paths, it must have SNMP enabled - even if the device has not been discovered by NNM.

➤ Paths in Path View may be different from connectivity displayed in Node View or Neighbor View.

In order for Path View to be effective, the following must be true:

- All layer 2 devices in the path are present in the Extended Topology database.
- All layer 3 devices, including switches, are accessible to SNMP queries from the management station.

Use Extended Topology to Compute Paths

New configuration parameters can be added to enhance the accuracy of path computation. These options only apply when Extended Topology is enabled. These options are disabled by default.

To enable the use of the new configuration parameters in the computation of paths in Path View, complete the following steps:

1 Make a backup copy of the following file:

- *UNIX*:

  `$OV_AS/webapps/topology/WEB-INF/web.xml`

- *Windows*:

  `%OV_AS%\webapps\topology\WEB-INF\web.xml`

2 Open the `web.xml` file for editing.

3 Find the following tag in the `web.xml` file:

  `PathServlet`

4 Within the context of the `PathServlet` tag, find the following parameter in the `web.xml` file:

  `restrictToET`

5 Set the value for `restrictToET` to `true` to use Extended Topology information for paths in Path View.

6 Stop and restart the ovas process to activate your changes. See Stop and Restart Processes on page 17 for more information.

Configure Extended Topology Path Algorithms

To make configuration changes to the Extended Topology algorithms for Path View, complete the following steps:

1 Make a backup copy of the following file:

- *UNIX*:

  `$OV_CONF/nnmet/topology/NMActiveRoute.Conf`

- *Windows*:

  `%OV_CONF%\nnmet\topology\NMActiveRoute.Conf.`

2 Open the `NMActiveRoute.Conf` file for editing.

3 Change the parameter value for the option you want to modify. The following parameters can be configured to tune the Extended Topology path algorithms:

- useSNMPConfDB

  This option defines the use of name resolutions from snmpConfDB. By default, this option is enabled. You should disable this option if you are confident that Extended Topology has updated information for most devices.

- useSubnetBasedPath

  This option modifies the subnet connectivity derivation. By default, this option is enabled. If you disable this option, the information retrieved may be more dynamic , but performance will decrease as SNMP queries increase.

- crossSwitchRouters

  When a layer 2 path is calculated, this parameter causes Path View to cross a device that has both switch and router capabilities. By default, this option is enabled. When this option is enabled, a router with switching capability is treated as a layer 2 device, and the path is computed with data from the Extended Topology database. SNMP queries are made for layer 3 devices. If you disable this option, you can achieve better results in meshed networks.

Path Analysis with Problem Diagnosis

Problem Diagnosis is an automated IP network path analysis tool that presents end-to-end path information accurately. Furthermore, Problem Diagnosis lets you see detailed information from nodes and devices in a particular path.

Problem Diagnosis gives NOC and IP Network Operators and Engineers tools for fast problem diagnosis and resolution in IP-based networks. In particular, Problem Diagnosis offers a probe-based path tool that finds and monitors the paths between itself and any reachable node that it is configured to test. A Problem Diagnosis probe collects data over time and generates statistical and usage data about the paths it monitors.

With Problem Diagnosis, you see path maps and tables based on topology data supplied by Problem Diagnosis probes. Probes can reside on any system in the network and can be configured to collect path information to any reachable destination in the network.

Major Components

Problem Diagnosis has three primary components:

- The Web-Based Graphical User Interface

  You launch the Problem Diagnosis view from Home Base. The Problem Diagnosis view is simple to use and presents data in easy-to-understand ways. From the Problem Diagnosis view, Problem Diagnosis opens its own windows for you to work in.

- The Problem Diagnosis Server

  The Problem Diagnosis server is the heart of the system. It assimilates information and responds to requests from a user running a Problem Diagnosis view.

  The Problem Diagnosis server gets its topology data from NNM, Problem Diagnosis probes, and other HP OpenView applications. Based on the topology data it mines, the Problem Diagnosis server can present several ways to examine the path(s) between nodes.

- The Problem Diagnosis Probes

  The Problem Diagnosis probes are key suppliers of data to the Problem Diagnosis system. Probes are independent Java applications that can reside on any supported system (see the *Release Notes* for supported platforms and versions). There is no limit to the number of probes you can install or the number of paths a probe can monitor. Likewise, a probe can be used by more than one Problem Diagnosis server.

  A probe collects information about paths between itself and any target. The probes use a technique similar to the familiar `traceroute` utility and run periodically to test the route to its target(s). On each run, a probe collects data about the following:

  - Devices along the route
  - Lag time between devices
  - New routes to the target

  When you request probe data, the Problem Diagnosis server contacts the probe for current data so that you see the most up-to-date information.

Table 13 provides a list of Problem Diagnosis features and limitations.

Table 13 Summary of Problem Diagnosis Features

Can	Cannot
• Show a path from where the probe resides to any reachable destination (firewalls can make some hosts unreachable). • Show path utilization statistics. • Show current ping rates. • Show baseline of ping rates. • Send events to subscribers (such as HP OpenView Operations) when a path changes. • Show status of hops using ping. • Send brownout events to the Alarm Browser, indicating a performance problem. • Show IP-layer 2 devices NOTE: After Problem Diagnosis has determined path data, the destination does not have to be currently up (or reachable) to see the historical paths.	• Show a path from any two arbitrary endpoints. • Show outbound interfaces (from ping's perspective) except on the probe's host.

See the online help for more details about retrieving information collected by Problem Diagnosis.

Start a Problem Diagnosis View

After it is installed and configured, Problem Diagnosis is very simple to operate. To start a Problem Diagnosis view, complete the following steps:

1 If it is not already running, start the Problem Diagnosis server with which you will connect. See Start and Stop a Server on page 219 for more information.

2. If they are not already running, start all probes for the server. It may take a few minutes before Problem Diagnosis has access to all probes. See Start and Stop a Probe on page 219 for instructions.
3. Launch Home Base. See Launch Home Base on page 86 for more information. Select the Problem Diagnosis view from Home Base.

Install Remote Probes

When you install Network Node Manager and set up Extended Topology, a Problem Diagnosis probe and server is automatically installed on the NNM management station. The installed probe is assigned to serve the installed Problem Diagnosis server.

Additional Problem Diagnosis probes can be installed throughout your network on supported operating systems. Each installed probe is assigned to serve at least one Problem Diagnosis server. A probe can be used by multiple Problem Diagnosis servers, and a Problem Diagnosis server can use multiple probes. For more information, see Link Servers and Probes on page 212.

Consider installing probes in key locations in the network where they can monitor crucial paths. There is an automated installation package in the installed NNM product to guide you through the installation process of remote probes.

Table 14 outlines the procedures necessary to install a Problem Diagnosis probe.

Table 14 Instructions for Installing Remote Probes

Platform of Probe System	**Instructions**

HP-UX	1 From the Problem Diagnosis server system, enter the following from a command window prompt to move to the appropriate directory:
	• *UNIX*:
	`cd $OV_MAIN_PATH/pdAE/bin`
	• *Windows*:
	`cd %OV_MAIN_PATH%\pdAE\bin`
	2 FTP `probeHP.tar` to the root directory of the system where the remote probe is to be installed.
	3 From the remote probe system, as user root, use the following command to untar `probeHP.tar` in the root directory:
	`tar -xvf probeHP.tar`
	4 As user `root`, enter the following command to install the probe:
	`./opt/OV/bin/pdAE/bin/pdpinstall.sh`
	The probe software is installed at the following location:
	`/opt/OV/bin/pdAE`
	You are prompted for the name of the Problem Diagnosis server with which the probe will communicate. Enter the fully-qualified DNS name of the assigned server.

Table 14 Instructions for Installing Remote Probes

Platform of Probe System	Instructions

Solaris	1 From the Problem Diagnosis server system, enter the following from a command window prompt to move to the appropriate directory:
	• *UNIX*:
	`cd $OV_MAIN_PATH/pdAE/bin`
	• *Windows*:
	`cd %OV_MAIN_PATH%\pdAE\bin`
	2 FTP `probeSUN.tar` to the root directory of the system where the remote probe is to be installed.
	3 From the remote probe system, as user `root`, use the following command to `untar probeSUN.tar` in the root directory:
	`tar -xvf probeSUN.tar`
	4 As user root, enter the following command to install the probe:
	`./opt/OV/bin/pdAE/bin/pdpinstall.sh`
	The probe software is installed at the following location:
	`/opt/OV/bin/pdAE`
	You are prompted for the name of the Problem Diagnosis server with which the probe will communicate. Enter the fully-qualified DNS name of the assigned server.

Table 14 Instructions for Installing Remote Probes

Platform of Probe System	Instructions

Windows	1 From the Problem Diagnosis server system, enter the following from a command window prompt to move to the appropriate directory:
	• *UNIX*:
	`cd $OV_MAIN_PATH/pdAE/bin`
	• *Windows*:
	`cd %OV_MAIN_PATH%\pdAE\bin`
	2 FTP `probeWIN.zip` to the root directory of the system where the remote probe is to be installed.
	3 From the remote probe system, unzip `probeWIN.zip` in the root directory. This unpackages the zip file to the following location:
	`C:\Program Files\HP OpenView.`
	4 Double-click the following file to install the probe:
	`C:\Program Files\HP OpenView\pdAE\bin\pdpinstall.vbs.`
	The probe software is installed at the following location:
	`C:\Program Files\HP OpenView\pdAE.`
	You are prompted for the name of the Problem Diagnosis server with which the probe will communicate. Enter the fully-qualified DNS name of the assigned server.

The Problem Diagnosis probe starts immediately and is configured to run automatically at system startup.

Configure Problem Diagnosis

A configuration XML file, `pdconfig.xml`, is created on the Problem Diagnosis server when Problem Diagnosis is installed at the following location:

- *UNIX*:

  ```
  $OV_MAIN_PATH/pdAE/config/pdconfig.xml
  ```

- *Windows*:

  ```
  %OV_MAIN_PATH%\pdAE\config\pdconfig.xml
  ```

You can use the configuration XML file to change your Problem Diagnosis configuration as follows:

- Add probes to the list of probes that the Problem Diagnosis server knows about. See Link the Server to a Probe on page 212.
- Notify servers that a probe exists. See Link the Probe to a Server on page 214.
- Tune brownout parameters. See Configure Brownouts on page 215.
- Change the server and probe ports used by Problem Diagnosis. See Configure Server and Probe Ports on page 217.

To make configuration changes, you need to edit the `pdconfig.xml` file. Be cautious and double-check the changes you make. Mistakes in the configuration file can cause the Problem Diagnosis server or probe to not work correctly.

If non-ASCII characters are added to XML files, you must preserve the UTF-8 codeset for these characters.

The Windows Notepad editor allows files to be saved in the UTF-8 code set. On many UNIX platforms, the iconv command can be used to translate from Shift JIS (or any other code set) into UTF-8 to make editing in a non-UTF-8 codeset simpler.

HP strongly recommends that you make a backup copy of the XML file before you edit the file to allow you to easily recover if necessary.

Link Servers and Probes

As you install a Problem Diagnosis probe, you assign it to a Problem Diagnosis server. The two transparently establish the configuration necessary to communicate, and the probe automatically shows up in the probe list on the server. In addition, a Problem Diagnosis server can get path data from any probe if the server is configured to know about the probe.

There are two ways to establish communications between a server and a probe that was not initially assigned to the server.

- You can add the probe to the list of probes the server knows about. This method is most useful when you want a server to know about several probes from which it could draw data. This approach is covered in Link the Server to a Probe on page 212.
- You can configure the probe to notify the server of its existence and let the server reconfigure itself. This method is most useful when you want a probe to know about several servers to which it can provides data. This approach is covered in Link the Probe to a Server on page 214.

Link the Server to a Probe

There are two steps to configure the server to contact a probe that was assigned to a different server when it was installed:

1. Manually edit the Problem Diagnosis configuration XML file on the Problem Diagnosis server to define which probe(s) the server can use. This file is located as follows:
    - *UNIX*:

      ```
      $OV_MAIN_PATH/pdAE/config/pdconfig.xml
      ```

    - *Windows*:

      ```
      %OV_MAIN_PATH%\pdAE\config\pdconfig.xml
      ```

    The configuration is created when a probe is installed and assigned to a server. If there is no configuration file on your system, the installation has not yet taken place.

The configuration file initially contains the following:

```
<?xml version= "1.0" encoding="UTF-8" ?>
<PD_CONFIG>
   <PROBELIST>
      <PROBE>
         <HOST_NAME> Icarus.naucrates.com </HOST_NAME>
         <IP_ADDRESS> 15.2.116.83 </IP_ADDRESS>
      </PROBE>
   </PROBELIST>
</PD_CONFIG>
```

The four lines in bold text create a "Probe Definition," which identifies `Icarus` as the host for a probe that the server can use. To add more probes, add more Probe Definitions. Probe Definitions must fall between the `<PROBELIST>` and `</PROBELIST>` tags, and they cannot overlap.

The following example shows the addition of a second probe, `Daedalus`:

```
<?xml version= "1.0" encoding="UTF-8" ?>
<PD_CONFIG>
   <PROBELIST>
      <PROBE>
         <HOST_NAME> Icarus.naucrates.com </HOST_NAME>
         <IP_ADDRESS> 15.2.116.83 </IP_ADDRESS>
      </PROBE>
      <PROBE>
         <HOST_NAME> Daedalus.naucrates.com </HOST_NAME>
         <IP_ADDERSS> 15.2.116.72 </IP_ADDRESS>
      </PROBE>
   </PROBELIST>
</PD_CONFIG>
```

2 After saving the changes to the configuration file, stop and restart the server to activate the changes. See Start and Stop a Server on page 219 for more information. Allow several minutes for the probe and server to synchronize.

When you access the server, you can now choose to use either `Icarus` or `Daedalus` as the probe.

Link the Probe to a Server

In most cases, it is not a good idea to have more than one Problem Diagnosis server configured for a probe. Since there is no way for one Problem Diagnosis server to talk to another Problem Diagnosis server, it could be difficult to obtain the path information you request.

There are two steps to configure a probe to notify an additional server of its existence:

1 Manually edit the probe configuration file. The probe configuration file resides on the probe system and is located as follows:

- *UNIX*:

  ```
  $OV_MAIN_PATH/pdAE/config/npprobe.conf
  ```

- *Windows*:

  ```
  %OV_MAIN_PATH%\pdAE\config\npprobe.conf
  ```

The probe configuration file looks like this:

```
SERVER=Minotaur.naucrates.com
SERVER_IP=12.12.121.212
SERVER_PORT=8068
```

The two lines in bold text identify the Problem Diagnosis server that is currently specified for this the probe. When the probe initializes, it notifies the server specified here of its existence. If necessary, the server adds the probe to its list of known probes.

To configure the probe to notify an additional server of its presence, change the bold lines in the probe configuration file to identify another server. Do not add additional lines. Simply change the existing lines to identify another server. The following example shows the contents of the configuration file with the addition of a server call `Aeolus`. Notice that the bolded lines now reference the new server name and new IP address.

```
SERVER=Aeolus.naucrates.com
SERVER_IP=12.12.112.121
SERVER_PORT=8068
```

A probe can only communicate with servers that use the port defined by that probe, `8068` in this example. If you change the port used by a server, you have to reconfigure each probe that the server uses to use the new port. See the Configure the Server Port on page 217 for instructions on changing the server port.

2 After saving the changes to the configuration file, stop and restart the probe to activate the changes. See Start and Stop a Probe on page 219 for more information. Allow several minutes for the probe and server to synchronize. Because `Aeolus` has no previous knowledge of the probe, it adds the probe to its `pdconfig.xml` file.

After the probe is added to the list of known probes on a server, it remains on that list even if you repeat this process to add it to yet another server.

Configure Brownouts

When Problem Diagnosis performs a trace from a probe to a destination, the packet round trip time to the destination is noted. If the packet time exceeds a calculated threshold, Problem Diagnosis pings the destination one time every minute for fifteen minutes, by default. Each packet round trip time is noted and compared to the threshold. When eight or more of the fifteen times exceed the threshold, a brownout event is generated.

Problem Diagnosis determines the point where a brownout is likely by looking at the nodes along a path from the probe to the destination. When the time for a hop from one node to the next exceeds its calculated threshold, Problem Diagnosis assumes that a spike is occurring between that hop and the previous hop.

A brownout event is sent to the Alarm Browser containing the probe, the destination, and the two hops where the spike occurred.

To change the way brownout events are generated, you can modify the Problem Diagnosis configuration file, `pdconfig.xml`, located as follows:

- *UNIX*:

```
$OV_MAIN_PATH/pdAE/config/pdconfig.xml
```

- *Windows*:

```
%OV_MAIN_PATH%\pdAE\config\pdconfig.xml
```

The Problem Diagnosis configuration file enables you to tune the brownout parameters described in Table 15.

Table 15 Brownout Parameters in Problem Diagnosis Configuration File

Tunable Parameter	Description	Default Value
BROWNOUT_INTERVAL	The time in milliseconds that Problem Diagnosis waits before generating another brownout event for a probe and destination combination.	86400000
BUCKET_SIZE	The number of measurements Problem Diagnosis stores for a particular hop from a particular path to a destination. After the value is reached, the oldest measurement is replaced by the newest measurement. NOTE: It is recommended that you do not change this parameter, especially after data has been collected.	24
BROWNOUT_NUM_SAMPLES	The number of times that Problem Diagnosis will attempt to ping the destination device. The frequency of attempts is one time per minute.	15
BROWNOUT_NUM_DEVIATIONS	The number used to calculate the threshold for brownout events. The formula for determining the threshold is the `BROWNOUT_NUM_DEVIATIONS` times the square root of the mean plus the mean.	3
BROWNOUT_BAD_SAMPLES	The number of times that the ping round trip time can exceed a threshold before a brownout event is generated.	8

Configure Server and Probe Ports

Ports 8068 and 8067 are used by the Problem Diagnosis server to communicate internally (8068) and with probes (8067). If either of these ports has been taken by other software, the ports used by the server must be changed. In the case of port 8068, probes must be reconfigured to reflect changes to the server.

Configure the Server Port

➤ Any probe accessed by a server that does not use port 8068 must be reconfigured to accommodate the different port number. If the probe is given a different port number, the probe's data will not be available to any server that uses a different port.

To change the Problem Diagnosis server port number from `8068` to a different port number, complete the following steps:

1. Stop the Problem Diagnosis server and stop all probes assigned to that server. See Start and Stop a Server on page 219 and Start and Stop a Probe on page 219.
2. Edit the configuration XML file, `pdconfig.xml`. Look for the following entry:

   ```
   <HTTP_PORT>8068</HTTP_PORT>
   ```

   Replace `8068` with a port number that is not used by any other networking software.
3. For all relevant Problem Diagnosis probes, edit `<`*`Install_directory`*`>/config/npprobe.conf` and replace `8068` with the same port number you used in the previous step.
4. Restart the server and restart all reconfigured probes.

Configure the Probe Port

To change the Problem Diagnosis probe port number from `8067` to another port number, complete the following steps:

1. Stop the Problem Diagnosis server. See Start and Stop a Server on page 219.

2 Edit the configuration XML file, `pdconfig.xml`. For each `<PROBE>` entry, edit the entry the following entry:

```
<HTTP_PORT>8067<HTTP_PORT>
```

Replace `8067` with a port number that is not used by any other networking software.

3 On each probe system, edit the following file:

- *UNIX*:

  ```
  /etc/services
  ```

- *Windows*:

  ```
  %windir%\system32\drivers\etc\SERVICES
  ```

Replace the value of the `netpath_http` entry with the new port number.

4 Restart the server.

Configure Probes

Problem Diagnosis includes a web-based probe configuration utility. To configure a probe, start the probe configuration utility as follows:

1 Start the probe to be configured. See Start and Stop a Probe on page 219 for more information.

2 Start the Problem Diagnosis server that uses the probe to be configured. See Start and Stop a Server on page 219 for more information.

3 Launch Home Base. See Launch Home Base on page 86 for more information.

4 Open the Problem Diagnosis view. See Access Dynamic Views on page 104 for more information.

5 Click **Configure** to open the Problem Diagnosis Configuration window.

6 From the Problem Diagnosis Configuration window, set probe targets and collection parameters by completing the following steps:

   a In the Select Probe list, choose the location of the probe you want to configure. If the list is empty, no probes have been installed.

   b Click the **Add** button to create an entry in the targets list.

      Double-click in the **Target** field, and add the fully-qualified name of a destination node. Problem Diagnosis will trace the paths to this node. Edit other fields in the entry as necessary.

   c After you have added all your targets, click **Apply** or **OK**. Both buttons record your changes.

See the Problem Diagnosis Probe Configuration topic in the online help for more details on configuring probes.

Other Administrative Tasks

This section describes other administrative tasks that you may need to perform periodically.

Start and Stop a Server

To start and stop the Problem Diagnosis server, you need to start and stop the `pd` process. See Stop and Restart Processes on page 17 for more information. This action also starts and stops the probe on the Problem Diagnosis server.

Do not force the termination of the Java process (`kill -9` on UNIX operating systems or `End Process` from the Task Manager on Windows operating systems). Forcing the termination may cause irrevocable corruption of the Problem Diagnosis data.

Start and Stop a Probe

Problem Diagnosis probes start immediately after installation and are configured to run automatically at system startup.

The commands for starting and stopping a Problem Diagnosis probe differ according to the platform on which the probe runs. Note that on Windows operating systems, probes are installed as a Windows service.

Platform	**Instructions**
UNIX	To stop a probe, execute: 1 `cd $OV_MAIN_PATH/pdAE/bin` 2 `./pdcentral.sh -stopProbe` To start a probe, execute: 1 `cd $OV_MAIN_PATH/pdAE/bin` 2 `./pdcentral.sh -startProbe`
Windows	To stop a probe service, execute: 1 Right-click on the `My Computer` desktop icon and select the `Manage` menu item. 2 In the navigator pane, double-click `Services and Applications`. 3 Double-click `Services`. 4 In the details pane, select the `NetPath` service, and click `Stop` in the applet toolbar. To start a probe service, execute: 1 Right-click on the `My Computer` desktop icon and select the `Manage` menu item. 2 In the navigator pane, double-click `Services and Applications`. 3 Double-click `Services`. 4 In the details pane, select the `NetPath` service, and click `Start` in the applet toolbar.

Troubleshooting Problem Diagnosis

Known Limitations

The limitations and special behaviors in Problem Diagnosis are covered in this section.

General Notes

- In some circumstances, the Problem Diagnosis server triggers a stack trace (visible in the Java console and/or the server error log) when you exit the browser. This exception is not fatal and is ignored. The operation of the server is not affected.
- Sometimes a DNS name resolves to multiple hosts, as in the case of a web farm. To improve performance, web farms and similar scenarios distribute incoming requests to one of an array of hosts, always choosing the least-busy host. This can be an issue for Problem Diagnosis.

  To resolve this issue, configure a probe with a destination for each web server and use each web server's IP address rather than a defining a single destination using the DNS address.
- *X-Windowing*: Severe display problems (including a complete failure to display the applet) have been observed when using X-windowing programs to redirect the Java applet display from UNIX® systems to Windows systems.
- *Windows systems only:* If HP VPIS is installed on the same system as NNM, you may have problems with Problem Diagnosis. A symptom of this conflict is an error message similar to the following message (depending on your JRE):

  The procedure entry point ??1List@@UAE@XZ cannot be located in the dynamic link library ovutil.dll

  To solve this problem, complete the following steps:

  a. Right-click on the **My Computer** desktop icon and select the **Properties** menu item.

  b. Select the **Advanced** tab.

  c. Click the **Environment Variables** button.

d. Find the system variable named Path. Edit the value for Path so that the NNM bin directory (C:\Program Files\HP OpenView\nnm\bin) comes *before* the VPIS bin directory (C:\rpmtools\bin).

e. Click **Apply**, then **Close**.

Install Notes

The installer cannot use the JRE from Symantec. You must have a different JRE in the **PATH** ahead of the Symantec JRE.

Problem Diagnosis Probe Notes

- When you first install a probe, the associated server should be running. If the service is not running, and if it is not started within an hour of the probe's installation, it may take up to an hour after the server is started for it to establish communications with the probe. You can eliminate this lag by stopping and restarting the probe after starting the server.
- A default gateway must be permanently configured for any system that hosts a probe. If the default gateway is not set, you will see routing loop error messages when no such loop exists.
- Path requests where the probe and target endpoints are the same device (for example, from `rantrave.diatribe.com` to `rantrave.diatribe.com`) have unpredictable results.
- *Windows only*: On some systems, running the probe as a service can cause the screen saver to stop working. This can be a security hazard if the computer will not lock when it is unattended. If this happens, you may want to run the probe in standalone mode rather than as a service.
  - To remove the probe from the Windows service, use the following commands from a command window prompt:
    - `cd %OV_MAIN_PATH%\pdAE\bin`
    - `pdcentral.bat -uninstall`
  - To run the probe in standalone mode, issue the following commands from a command window prompt:
    - `cd %OV_MAIN_PATH%\pdAE\bin`
    - `pdcentral.bat -startProbeNoSvc`

Java Behaviors

There are some minor instabilities in Java behaviors that you should be aware of. Below are some examples. With the exception of the `CLASSPATH` conflict issue, the workarounds are very simple.

- The following can occur on Windows operating systems when the Display settings have the "Show window contents while dragging" option turned on.

  If you drag the Main Dialog window after you click **Go**—but before the path list appears—the resulting path list is obscured. It is actually present, but you have to drag the bottom border of the window down to see it.

- The **Go** and **Stop** buttons in the Main Dialog window may randomly change colors. This problem has only been seen on Windows operating systems, and the culprit is actually the video display driver. If you switch to another Display:Settings:Color Palette value, the problem should disappear.

- After you click **GO**, the button is disabled until the results of your request are available. Depending on the Java plug-in and platform you are using, you may or may not see a "Busy" cursor during this period to indicate that your request is being serviced. If the "Busy" cursor does not appear, you should assume that Problem Diagnosis is working and will shortly return the results of your request.

- *HP-UX only*: When you start the configuration applet, the window may not start correctly. It displays only a title bar and the Applet Window warning message. If this happens, drag the corner to enlarge the window, and the content will display correctly.

- In some situations, your system's Java `CLASSPATH` environment variable can cause problems. Typical symptoms of a `CLASSPATH` conflict are as follows:

  — The Problem Diagnosis application window fails to launch and the browser status line indicates "`Applet not initiated`".

  — Netscape throws errors when you try to launch the online help, and the navigation panel in the help window is blank.

If you see these symptoms (or other unexplained failures in the Java interfaces), run Problem Diagnosis in a shell with an empty `CLASSPATH` as follows:

Platform	**Instructions**
UNIX	• Open a new terminal window. • Type: `export CLASSPATH=""` • Type: `netscape &` • Start Problem Diagnosis from the resulting browser as usual. See Start a Problem Diagnosis View on page 206.
Windows	• Open a command window. • Type the following command: `set CLASSPATH=""` • Start Problem Diagnosis by launching your browser from this command window. Under default installations, the commands to launch Netscape and Internet Explorer are: — Netscape: `"C:\Program Files\Netscape\Communicator\Program\netscape.exe"` — Internet Explorer: `"C:\Program Files\Plus!\Microsoft Internet\iexplore.exe"` See Launch Home Base on page 86 for information on launching views from a command window.

- On UNIX systems, there are conditions under which the Problem Diagnosis applets and/or online help will not launch correctly in Netscape. If you have trouble of this kind, follow these steps:

  a Exit from Netscape.

  b Enter: `export MOZILLA_HOME=<`*`Netscape_Installation_Directory`*`>`

  c Restart Netscape.

  d Restart Problem Diagnosis.

Clean Up a Corrupt Database

The Problem Diagnosis database contains all of the path data collected from the Problem Diagnosis server and all of the remote probes. If the Problem Diagnosis database is in a corrupt state and cannot be recovered, clean out the contents of the Problem Diagnosis database directory with the following steps:

1. Stop the `pd` process on the NNM management station. See Stop and Restart Processes on page 17 for more information.
2. Go to the database directory on the NNM management station:
   - *UNIX*:

     ```
     $OV_MAIN_PATH/pdAE/database
     ```
   - *Windows*:

     ```
     %OV_MAIN_PATH%\pdAE\database
     ```
3. Remove all the files from the database directory.

When you remove the database files from the Problem Diagnosis database directory, you delete all of your stored Problem Diagnosis data. This includes all of the data collected by the Problem Diagnosis probes.

Check the Status of a Problem Diagnosis Server

Typically, you would check the status of an NNM component by executing `ovstatus`. However, sometimes the output from executing `ovstatus pd` indicates that the Problem Diagnosis server is up and running when it is not.

The best way to check the status of the Problem Diagnosis server is to check the connection to the Problem Diagnosis database. If a connection to the database can be established, then the Problem Diagnosis server is up and running.

To connect to the database and verify that the Problem Diagnosis server is running, execute:

```
pdcentral.sh -dbmgr
```

Check the Status of a Remote Probe

To determine whether a remote Problem Diagnosis probe is running, check to see if the Java process is running on the system hosting the probe with the following steps:

1 Run the following command:

```
ps -ef | grep java
```

2 In the output of the command, look for the following process (or similar):

```
/opt/OV/jre/jre1.4/bin/PA_RISC/java
com.hp.ov.pd.netpath.NPP
```

If you see the process, the Problem Diagnosis probe is running.

7 Syslog Integration

Introducing the Syslog Functionality

The Syslog Integration functionality enables the management of network equipment from syslog messages. Certain types of network equipment do not have SNMP traps or supporting MIBs for all error and warning conditions. For operators who need to manage these conditions, the Syslog Integration functionality provides the ability to map syslog messages to SNMP traps for presentation or analysis.

Network Node Manager includes out-of-the-box conditions for which syslog messages are mapped to SNMP traps. You can add new conditions through the NNM Syslog Trap Mapping Configuration interface or the HP OpenView Operations message source template configuration windows, depending on your deployment mode.

Syslog Integration functionality is available with Network Node Manager (NNM) Advanced Edition. Syslog Integration is also available with HP OpenView Operations 8.0 with NNM. Syslog Integration must be configured on an NNM management station running an UNIX® operating system.

Syslog Integration Deployment Options

There are two deployment options available when you use the Syslog Integration functionality.

- NNM standalone
- OpenView Operations (OVO) with NNM

➤ For the OVO with NNM deployment option, two scenarios are allowed. The Syslog functionality resides on a NNM management station (co-located with OVO) or the Syslog Integration functionality resides on a remote NNM management station (not co-located with OVO).

NNM Standalone Configuration

The NNM standalone configuration consists of deploying an HP OpenView Operations (OVO) agent on the NNM management station. In short, the embedded OVO agent uses preconfigured templates to parse incoming syslog messages that match a certain pattern. The matched syslog messages are mapped to SNMP traps and forwarded to NNM to be displayed in the NNM Alarm Browser. See Figure 13 on page 229 for an illustration.

In this approach, the following components comprise the heart of the architecture:

- Embedded OVO Agent

  When Syslog Integration is configured, an OVO agent is installed on the NNM management station. Part of the embedded OVO agent is the log file encapsulator that is responsible for listening for syslog events from log files. The log file encapsulator filters and formats syslog events according to information in templates. The log file encapsulator then forwards relevant information in the form of messages to an NNM background process, syslogTrap, which maps the syslog messages into SNMP traps.

- NNM Management Station

  When Syslog Integration is configured, the NNM management station receives formatted syslog messages from the embedded OVO agent. syslogTrap, which is a background process on the NNM management station, maps the syslog messages to OpenView SNMP traps. This process registers with the OVO agent at the message stream interface and filters all messages of message type NNMsyslog.

  The events generated from syslogTrap are of type OV_Syslog and have the same general format:

  — A unique enterprise-specific ID to identify the syslog message.

  — An OpenView application name varbind identifying the source as syslogTrap.

  — A varbind identifying the source of the event.

  — Varbinds identifying the card/port/interface, status, and so on, as appropriate.

  — A varbind that encapsulates the original message.

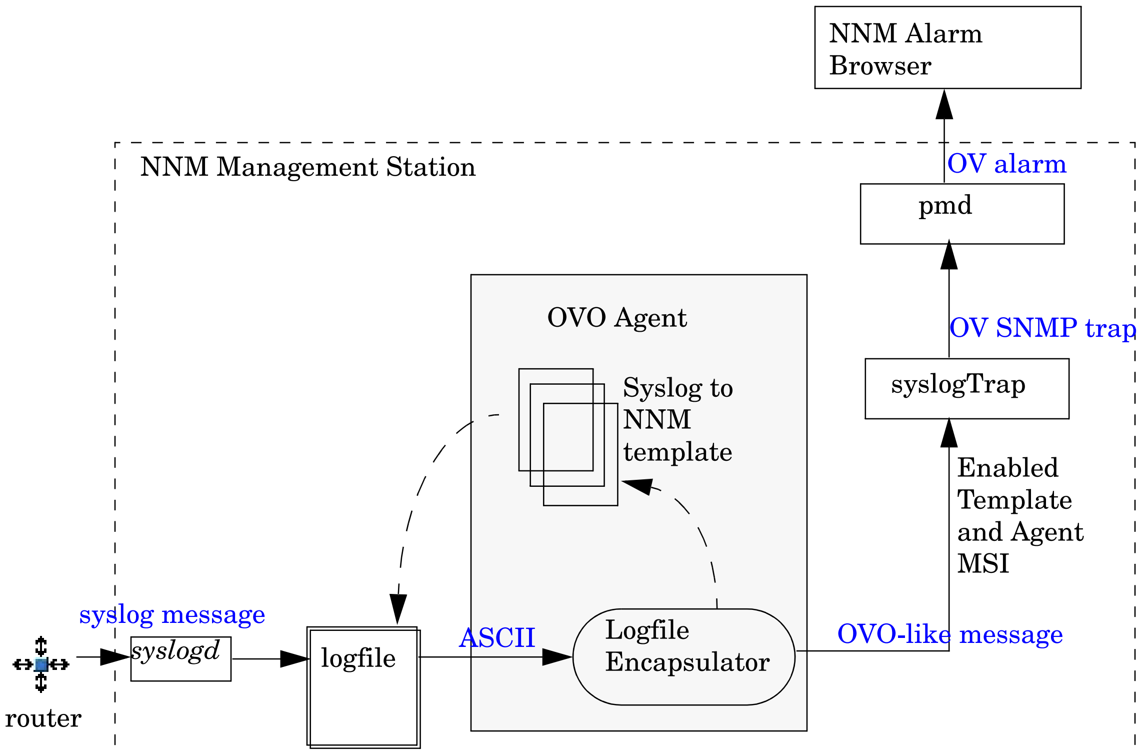
The SNMP traps are then sent to the NNM postmaster process, pmd, where the traps can participate in correlation or analysis. For example, when the NNM Smart Plug-in for LAN/WAN Edge is installed, it provides advanced correlators for frame relay traps and syslog messages. pmd forwards the formatted syslog messages to the NNM Alarm Browser.

- NNM Alarm Browser

  Processed syslog messages are displayed in the Status Alarm category of the NNM Alarm Browser.

Figure 13 shows the flow of events for the NNM standalone configuration. This illustration assumes that the router has been configured to forward syslog events to the NNM management station.

Figure 13 Flow of Events for NNM Standalone Configuration



To modify the Syslog to NNM template, use the NNM Syslog Trap Mapping Configuration Interface on page 232.

OpenView Operations with NNM Configurations

The OVO with NNM configuration option consists of deploying HP OpenView Operations with Network Node Manager. Two deployment approaches exist for this configuration option. First, the NNM management station on which the Syslog Integration functionality is installed resides on the OVO server. Optionally, the Syslog Integration functionality resides on a NNM management station separate from the OVO server system.

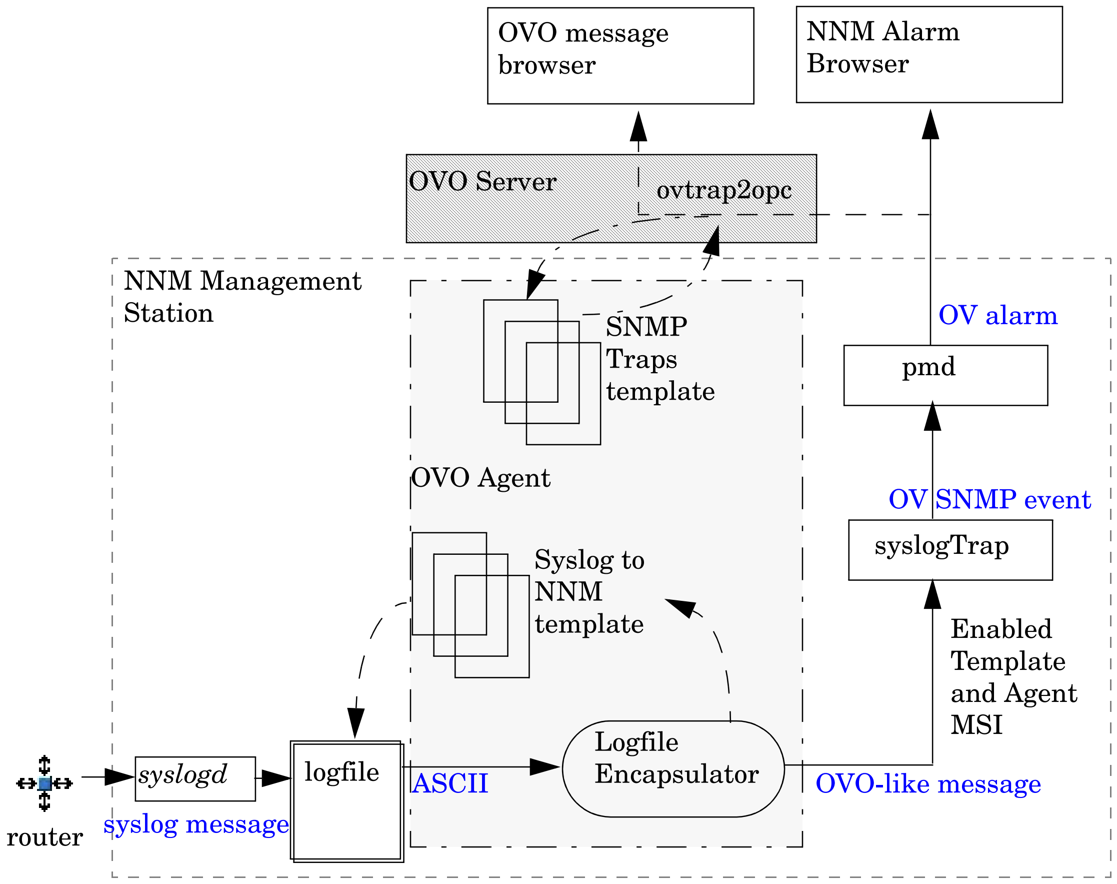
With either approach, you install an OVO agent on the NNM management station from the OVO server and use the OVO tools to configure the OVO agent to process syslog events.

This approach is similar to the NNM standalone configuration option. The architectural differences between the two deployment options are:

- The OVO agent is not embedded on the NNM management station; rather the OVO agent coexists with the NNM management station.
- The OVO server is used to start and configure the OVO agent on the NNM management station to process syslog messages. As an OVO administrator, you use the OVO configuration tools to upload the syslog message source templates to the OVO agent and synchronize the OVO message source templates with NNM trapd.conf.
- NNM forwards the SNMP traps to either (or both) the OVO message browser or the NNM Alarm Browser, depending on how messages are configured to be diverted through the system.

Figure 14 on page 231 shows the flow of events for the OVO with NNM configuration. This illustration assumes that the router is configured to forward syslog events to the NNM management station.

Figure 14 Flow of Events for OVO with NNM Configuration



To construct and modify the Syslog to NNM message source template, use the standard OVO template editor. To launch the OVO template editor, see OVO Message Source Templates Window on page 232.

Configuration Tools

This section describes the tools needed to configure the NNM Syslog Integration functionality. The tools and steps required differ depending on your deployment option.

NNM Syslog Trap Mapping Configuration Interface

When you deploy the NNM standalone configuration option, you construct and modify the Syslog Integration message source template conditions through the Syslog Trap Mapping Configuration interface.

To launch the Syslog Trap Mapping Configuration interface, run the following command:

```
$OV_BIN/ovsyslogcfg
```

For instructions on how to use the Syslog Trap Mapping Configuration interface, see the *Syslog Trap Mapping Configuration* online help.

OVO Message Source Templates Window

When you deploy the OVO with NNM configuration option, you construct and modify Syslog Integration message source template conditions through the Message Source Templates configuration window.

To open the Message Source Templates configuration window, launch the OVO administrator interface ($OV_BIN/OpC/opc), and select **Window** → **Message Source Templates**.

For instructions on how to use the Message Source Templates configuration window, see the *OVO Administrator* online help.

Syslog Configuration Command

The Syslog Integration functionality is not enabled when NNM is installed. To enable and deploy a syslog configuration, run the following script:

```
$OV_BIN/setupSyslog.ovpl
```

Use the -help option to display a help message for the setupSyslog.ovpl command options.

In the NNM Standalone Deployment Mode

When the NNM standalone configuration option is deployed, execute the syslog configuration command with the -standalone option by typing the following:

```
$OV_BIN/setupSyslog.ovpl -standalone
```

This command does the following on your NNM management station:

- Deploys the out-of-the-box syslog template, Syslog to NNM.
- Activates the embedded OVO agent.
- Activates and registers the SNMP mapping background process, syslogTrap.

For detailed configuration steps, see Configuring Syslog Integration for NNM Standalone on page 238.

The setupSyslog.ovpl configuration command is also used to deploy new syslog configurations. After editing syslog message source template conditions with the Syslog Trap Mapping Configuration interface, execute setupSyslog.ovpl -standalone -deploy to deploy the new configuration. The -deploy command option generates and encrypts the new template and restarts the embedded OVO agent so that the new template is reloaded.

In the OVO with NNM Deployment Mode

When the OVO with NNM configuration option is deployed, execute the syslog configuration command with the -server option by typing the following command:

```
$OV_BIN/setupSyslog.ovpl -server
```

This command creates an uploadable template configuration directory for the out-of-the-box syslog template, Syslog to NNM.

After executing this command, you must perform the following configuration steps:

- Upload the Syslog to NNM template into the OVO database.
- Start and register the NNM mapping process, syslogTrap.
- Enable the template MSI and the OVO agent MSI.
- Assign and install the Syslog to NNM template to the OVO agent installed on the NNM management station.

For detailed configuration steps, see Configuring Syslog Integration for OVO with NNM on the Same System on page 239.

Default Syslog Trap Mappings

NNM includes out-of-the-box template conditions for which syslog messages are mapped to OpenView SNMP traps. Each type of syslog message to be mapped is defined in one template condition. The conditions are contained in the Syslog to NNM template.

The Syslog to NNM template includes out-of-the-box message condition definitions to match the following types of messages:

- Link Down and Link Up messages
- Line Protocol messages
- Frame Relay messages
- HSRP messages
- OSPF messages
- Trunk (port aggregation) messages, including Port Aggregation Protocol (PAGP) and Dynamic Trunking Protocol (DTP) messages
- Border Gateway Protocol (BGP) messages
- Spanning Tree Protocol (STP) message
- CDP messages

OpenView traps are defined to support the out-of-the-box syslog message mappings. The list of mapped traps is described in Table 16.

Table 16 Syslog Trap Mappings

Syslog Message	Event Generated
`%LINK-3-UPDOWN (down)`	`OV_Syslog_LinkDown`
`%LINK-3-UPDOWN (up)`	`OV_Syslog_LinkUp`
`%LINEPROTO-5-UPDOWN (down)`	`OV_Syslog_LineProtoDown`
`%LINEPROTO-5-UPDOWN (up)`	`OV_Syslog_LineProtoUp`
`%FR-5-DLCICHANGE (INVALID)`	`OV_Syslog_FrameDLCI_Invalid`
`%FR-5-DLCICHANGE (INACTIVE)`	`OV_Syslog_FrameDLCI_Inactive`
`%FR-5-DLCICHANGE (ACTIVE)`	`OV_Syslog_FrameDLCI_Active`

Table 16 Syslog Trap Mappings

Syslog Message	Event Generated
%OSPF-5-ADJCHG (DOWN)	OV_Syslog_OSPF_Neighbor_Down
%OSPF-5-ADJCHG (FULL)	OV_Syslog_OSPF_Neighbor_Up
%STANDBY-6-STATECHANGE (Speak)	OV_Syslog_HSRP_State_Speak
%STANDBY-6-STATECHANGE (Standby)	OV_Syslog_HSRP_State_Standby
%STANDBY-6-STATECHANGE (Active)	OV_Syslog_HSRP_State_Active
%STANDBY-6-STATECHANGE (Init)	OV_Syslog_HSRP_State_Init
%STANDBY-3-DUPADDR	OV_Syslog_HSRP_Duplicate_Address
%SNMP-5-MODULETRAP	OV_Syslog_Card_Up OV_Syslog_Card_Down
%SYS-5-MOD_NORESPONSE	OV_Syslog_Card_Failure
%SYS-5-MOD_OK	OV_Syslog_Card_Online
%SYS-5-MOD_REMOVE	OV_Syslog_Card_Removed
%SYS-5-MOD_INSERT	OV_Syslog_Card_Inserted
%SYS-5-MOD_RESET	OV_Syslog_Card_Reset
%SYS-3-MOD_FAIL	OV_Syslog_Card_Failure
%SYS-3-MOD_FAILREASON	OV_Syslog_Card_Failure
%SYS-3-MOD_CFGMISMATCH1	OV_Syslog_Card_Config_Mismatch
%SYS-3-MOD_CFGMISMATCH2	OV_Syslog_Card_Config_Mismatch
%SYS-3-MOD_CFGMISMATCH3	OV_Syslog_Card_Config_Mismatch
%SYS-3-MOD_CFGMISMATCH4	OV_Syslog_Card_Config_Mismatch
%PAGP-5-PORTTOSPT	OV_Syslog_PAGP_JoinedBridgePort

Table 16 Syslog Trap Mappings

Syslog Message	Event Generated
`%PAGP-5-PORTFROMSPT`	`OV_Syslog_PAGP_LeftBridgePort`
`%DTP-3-TRUNKPORTFAIL`	`OV_Syslog_Trunk_Port_Fail`
`%DTP-3-NONTRUNKPORTFAIL`	`OV_Syslog_Trunk_NonTrunkPort_Fail`
`%DTP-5-TRUNKPORTON`	`OV_Syslog_Trunk_Port_On`
`%DTP-5-TRUNKPORTCHG`	`OV_Syslog_Trunk_Port_Change`
`%OSPF` (all other OSPF messages)	`OV_Syslog_OSPF_Default_Message`
`%STANDBY` (all other HSRP messages)	`OV_Syslog_HSRP_Default_Message`
`%PAGP` (all other PAGP messages)	`OV_Syslog_PAGP_Default_Message`
`%SYS-`*n*`-MOD` (all other Cisco card messages)	`OV_Syslog_Card_Default_Message`
`%DTP` (all other %DTP trunk messages)	`OV_Syslog_Trunk_Default_Message`
`%SPANTREE`	`OV_Syslog_Spantree_Default_Message`
`%CDP`	`OV_Syslog_CDP_Default_Message`
`%BGP`	`OV_Syslog_BGP_Default_Message`

Configuring Syslog Integration

This chapter provides the steps necessary to setup the Syslog Integration functionality in either the NNM standalone deployment mode or OVO with NNM deployment mode.

Prerequisites for Configuring Syslog

DCE Software Requirements for Syslog on HP-UX Systems

Part of the configuration process for the Syslog integration includes installing an HP Operations (OVO) agent on the NNM management station. The embedded OVO agent requires that specific components are installed on HP-UX systems. If you did not complete this during your NNM install, refer to the Installation Guide for more information.

Syslog Requirement in an NIS Environment

➤ This step is required only for OVO with NNM configuration deployments.

If the NNM management station used for syslog monitoring is a Network Information Service (NIS or NIS+) client, you need to install the default OVO user, opc_op, and user group, opcgrp, on the NIS server.

If the default OVO user, opc_op, and user group, opcgrp, are not installed on the NIS server, or at least installed locally on the managed node, you may get errors when you install agents and agent software on the managed node.

To add opc_op locally on the managed node, edit the /etc/passwd file to include an entry for opc_op. For example, the entry in the /etc/passwd file could read:

```
opc_op:*:777:299:OpC default operator:/home/opc_op:/usr/bin/
ksh
```

Configuring Syslog Integration for NNM Standalone

➤ DCE software requirements must be installed before configuring the Syslog Integration functionality. See DCE Software Requirements for Syslog on HP-UX Systems on page 237.

The Syslog Integration functionality is not enabled when NNM Advanced Edition is installed. To enable and deploy a syslog configuration, run the following script:

```
$OV_BIN/setupSyslog.ovpl
```

To enable Syslog Integration for NNM standalone configurations, run the following command on the NNM management server:

```
$OV_BIN/setupSyslog.ovpl -standalone
```

This command does the following on your NNM management station:

- Deploys the out-of-the-box syslog template, Syslog to NNM.
- Installs and activates the embedded OVO agent.
- Activates and registers the SNMP mapping process, syslogTrap.

To customize the out-of-the-box syslog template, see Modifying the Syslog to NNM Template on page 238. For information about verifying your syslog configuration, see Testing Syslog Integration on page 245.

Modifying the Syslog to NNM Template

For customizations to the Syslog to NNM template, use the Syslog Trap Mapping Configuration interface. To launch the Syslog Trap Mapping Configuration interface, execute:

```
$OV_BIN/ovsyslogcfg
```

For instructions on how to use the Syslog Trap Mapping Configuration interface, see the *Syslog Trap Mapping Configuration* online help and NNM Syslog Trap Mapping Configuration Interface on page 232.

For more information about the Syslog to NNM template and its conditions, see Understanding the Syslog to NNM Template on page 251.

Configuring Syslog Integration for OVO with NNM on the Same System

⚠ If you are installing OVO on top of an NNM installation, be sure to disable Syslog Integration functionality prior to installing OVO. For instructions on disabling the Syslog Integration functionality, see Removing Syslog Integration on page 245.

The Syslog Integration functionality is not enabled when NNM is installed.

To enable and deploy a syslog configuration, do the following on the NNM management station:

1 Run the following command:

```
$OV_BIN/setupSyslog.ovpl -server
```

This command creates an uploadable template configuration directory for the out-of-the-box syslog template, Syslog to NNM, under /var/opt/OV/share/tmp/NNMsyslogTraps. This command also creates the syslog process, syslogTrap.

2 To register the syslogTrap process with NNM, complete the following steps:

a Verify that the syslogTrap process is not already registered. Open the $OV_CONF/ovsuf file and check to see if the syslogTrap process is listed. If the process is not already registered, proceed with the registration.

b Go to the $OV_LRF directory.

c Run the following commands:

```
$OV_BIN/ovaddobj
$OV_LRF/syslogTrap.lrf
```

d Verify that the process is registered. Open the $OV_CONF/ovsuf file and check that syslogTrap is listed.

3 Upload the Syslog to NNM template into the OVO database with the following command:

```
$OV_BIN/OpC/opccfgupld NNMsyslogTraps
```

4 As user root, start HP OpenView Operations with the following command:

```
$OV_BIN/OpC/opc
```

Since the NNM process, syslogTrap, relies on message stream interface (MSI) to map syslog messages to SNMP traps, the MSI must be enabled on the template and the OVO agent.

5 To enable the MSI on the Syslog to NNM template, complete the following steps:

   a To open the Message Source Templates window, select **Window** → **Message Source Templates**.

   b Select the Syslog to NNM template.

   c Click **Modify**.

   d Click **Advanced Options**.

   e Check Agent MSI and select **Copy Messages**.

   f Click **OK**.

6 To enable the MSI on the OVO agent coexisting on the NNM management station, complete the following steps:

   a From the Node Bank, select the node containing the OVO agent that coexists on the NNM management station.

   b Click **Actions** → **Node** → **Modify**.

   c From the Modify Node window, click **Advanced Options**.

   d From the Node Advanced Options window, check Enable Output from the Message Stream Interface pane.

   e Click **Close**.

   f Click **OK** from the Modify Node window.

7 To assign the Syslog to NNM template to the OVO agent that coexists on the NNM management station, complete the following steps:

   a From the Node Bank, select the node containing the OVO agent that coexists on the NNM management station.

   b Click **Actions** → **Agents** → **Assign Templates**. This opens the Define Configuration window.

   c Click **Add**. The Add Configuration window displays.

d Click **Open Template Window**.

e From the Message Source Templates window, select the Syslog to NNM template.

f From the Add Configuration window, click **Get Template Selections**.

g Click **OK** in the Add Configuration window.

h Click **OK** in the Define Configuration window.

8 Install the Syslog to NNM template on the OVO agent that coexists on the NNM management station by doing the following:

a From the Node Bank, select the node containing the OVO agent that coexists on the NNM management station.

b Click **Actions** → **Agents** → **Install/Update SW & Config**.

c In the Install/Update ITO Software and Configuration window, make sure the correct managed node is listed in the Target Nodes pane.

d Check Templates in the Components pane.

e Click **OK** to cause the template to be deployed on the managed node. After the template deploys, you should see a message in the OVO message browser indicating that the agent system has been updated.

9 To start the NNM mapper process, syslogTrap, run the following command:

```
$OV_BIN/ovstart syslogTrap
```

10 To pull in the latest SNMP trap definitions, complete the following steps:

- Run the following command:

  ```
  $OV_BIN/OpC/util/ovtrap2opc
  ```

  See the *ovtrap2opc* reference page for information about this command.

- Select **y** when prompted to upload to SNMP Traps template.

- Install the SNMP Traps template on the OVO agent that coexists on the NNM management station.

11 Test your syslog configuration by sending sample syslog messages through the system. For instructions, see Testing Syslog Integration on page 245.

Configuring Syslog Integration for OVO with NNM on Separate Systems

⚠ If you are installing OVO on top of an NNM installation, be sure to disable Syslog Integration functionality prior to installing OVO. For instructions on disabling the Syslog Integration functionality, see Removing Syslog Integration on page 245.

The Syslog Integration functionality is not enabled when NNM is installed.

To enable and deploy a syslog configuration, do the following on the NNM management station:

1. From the OVO server, run the following command:

   ```
   $OV_BIN/setupSyslog.ovpl -server
   ```

   This command creates an uploadable template configuration directory for the out-of-the-box syslog template, Syslog to NNM, under /var/opt/OV/share/tmp/NNMsyslogTraps. This command also creates the syslog process, syslogTrap.

2. From the OVO server, upload the Syslog to NNM template into the OVO database with the following command:

   ```
   $OV_BIN/OpC/opccfgupld NNMsyslogTraps
   ```

3. From the OVO server, run the following command:

   ```
   $OV_BIN/setupSyslog.ovpl -server -disable
   ```

4. From the remote NNM management station, run the following command:

   ```
   $OV_BIN/setupSyslog.ovpl -server
   ```

5. From the remote NNM management station, register the syslogTrap process with NNM with the following steps:

   a. Verify that the syslogTrap process is not already registered. Open the $OV_CONF/ovsuf file and check to see if the syslogTrap process is listed. If the process is not already registered, proceed with the registration.

   b. Go to the $OV_LRF directory.

   c. Run the following commands:

      ```
      $OV_BIN/ovaddobj
      $OV_LRF/syslogTrap.lrf
      ```

   d. Verify that the process is registered. Open the $OV_CONF/ovsuf file and check that syslogTrap is listed.

6. From the OVO server, as user root, start HP OpenView Operations with the following command:

   ```
   $OV_BIN/OpC/opc
   ```

   Since the NNM process, syslogTrap, relies on message stream interface (MSI) to map syslog messages to SNMP traps, the MSI must be enabled on the template and the OVO agent.

7. To enable the MSI on the Syslog to NNM template, complete the following steps:

   a. To open the Message Source Templates window, select **Window** → **Message Source Templates**.

   b. Select the Syslog to NNM template.

   c. Click **Modify**.

   d. Click **Advanced Options**.

   e. Check **Agent MSI** and select **Copy Messages**.

   f. Click **OK**.

8. To enable the MSI on the OVO agent coexisting on the NNM management station, complete the following steps:

   a. From the Node Bank, select the node containing the OVO agent coexisting on the NNM management station.

   b. Click **Actions** → **Node** → **Modify**.

   c. From the Modify Node window, click **Advanced Options**.

   d. From the Node Advanced Options window, check Enable Output from the Message Stream Interface pane.

   e. Click **Close**.

   f. Click **OK** from the Modify Node window.

9. To assign the Syslog to NNM template to the OVO agent that coexists on the NNM management station, complete the following steps:

   a. From the Node Bank, select the node containing the OVO agent that coexists on the NNM management station.

   b. Click **Actions** → **Agents** → **Assign Templates**. This opens the Define Configuration window.
   c. Click **Add**. The Add Configuration window displays.
   d. Click **Open Template Window**.
   e. From the Message Source Templates window, select the Syslog to NNM template.
   f. From the Add Configuration window, click **Get Template Selections**.
   g. Click **OK** in the Add Configuration window.
   h. Click **OK** in the Define Configuration window.

10. Install the Syslog to NNM template on the OVO agent that coexists on the NNM management station by doing the following:
   a. From the Node Bank, select the node containing the OVO agent that coexists on the NNM management station.
   b. Click **Actions** → **Agents** → **Install/Update SW & Config**.
   c. In the Install/Update ITO Software and Configuration window, make sure the correct managed node is listed in the Target Nodes pane.
   d. Check Templates in the Components pane.
   e. Click **OK** to cause the template to be deployed on the managed node. After the template deploys, you should see a message in the OVO message browser indicating that the agent system has been updated.

11. To start the NNM mapper process, syslogTrap, run the following command:

```
$OV_BIN/ovstart syslogTrap
```

12. To pull in the latest SNMP trap definitions, complete the following steps:
   - Run the following command:

     ```
     $OV_BIN/OpC/util/ovtrap2opc
     ```

     See the *ovtrap2opc* reference page for information about this command. Details about how to access reference pages are in the online help.
   - Select **y** when prompted to upload to SNMP Traps template.
   - Install the SNMP Traps template on the OVO agent that coexists on the NNM management station.

13 Test your syslog configuration by sending sample syslog messages through the system. For instructions, see Testing Syslog Integration on page 245.

Locating Syslog Log Files

On HP-UX operating systems, the location of the system log file is as follows:

```
/var/adm/syslog
```

On Solaris operating systems, the location of the system log file is as follows:

```
/var/adm/messages
```

On Linux 2.6 operating systems, the location of the system log file is as follows:

```
/var/log/messages
```

Testing Syslog Integration

Use the UNIX command line tool, `logger`, to write messages to the system log file. Read the reference page for more information on how to use the command. Details about how to access reference pages are in the online help.

For example, to create a *Line Protocol status Down* syslog entry, do the following:

HP-UX: logger %LINEPROTO-5-UPDOWN: Line protocol on Interface interface2, changed state to down

Solaris: logger -p user.err %LINEPROTO-5-UPDOWN: Line protocol on Interface interface2, changed state to down

When Syslog Integration is enabled, you should see syslog messages matching the conditions of the Syslog to NNM template forwarded to the NNM Alarm Browser or the OVO message browser, depending on your configuration.

Removing Syslog Integration

When you remove Network Node Manager from a system, Syslog Integration is not completely removed. The OVO agent is left enabled and running on the system.

To remove the remaining Syslog Integration components, complete the following steps:

1. Disable the Syslog Integration feature with the following command:
   - For NNM standalone configurations, type:

     ```
     $OV_BIN/setupSyslog.ovpl -standalone -disable
     ```

   - For OVO with NNM configurations, type:

     ```
     $OV_BIN/setupSyslog.ovpl -server -disable
     ```

2. For NNM standalone configurations, remove the OVO agent software from the NNM management station with the following steps:
   a. To open the SD Remove window, run the following command:

      HPUX:

      ```
      swremove
      ```

      Solaris:

      ```
      pkgrm
      ```

   b. Select the **ITOAgent** software package name from the Name list.
   c. Click **Actions** → **Remove** to remove the OVO agent from the NNM management station.

Customizing Message Source Templates

This section provides the steps necessary to create, modify, and enable Syslog Integration message source templates. The configuration tools used to modify message source templates are also described.

Overview of Message Source Templates

OVO agents are configured via message source templates. The OVO agent can only format and forward a message that is described in a message source template.

In the NNM standalone configuration, the OVO agent is embedded on the NNM management station. In the OVO with NNM configuration, the OVO agent coexists on the NNM management station. In either case, the OVO agent is configured to monitor the status of and collect information from syslog messages through the Syslog to NNM message source template.

Message source templates work by identifying strings within messages in message streams. When a message matches the conditions defined in the message source templates, it is processed according to the rules defined in the template. When the Syslog Integration functionality is enabled, messages matching markers defined in the Syslog to NNM template are forwarded to the NNM syslogTrap process. This process maps the syslog messages to SNMP traps.

Message source templates consist of the following elements:

- The type of message source from which you want to collect messages, such as a log file, a trap, an OVO message interface, or an action is included. In the case of the Syslog to NNM template, the message source is a log file.
- Message conditions and suppress conditions that match a set of attributes and define responses to received messages are included. These conditions filter incoming messages from the message source. The conditions also determine how the important messages are displayed in the operator window.
- Options, such as default message logging, are included.

Understanding the Template Configuration Tools

There are two main template editing tools used to create and modify syslog configuration template conditions. For NNM standalone configurations, use the Syslog Trap Mapping Configuration interface (see NNM Syslog Trap Mapping Configuration Interface on page 232). For OVO with NNM configurations, use the OVO Message Source Templates window (see OVO Message Source Templates Window on page 232). Details on how to use these template editing tools can be found in their respective online help topics.

NNM Syslog Trap Mapping Configuration Interface

When the NNM standalone configuration option is deployed, you construct and modify Syslog Integration message source template conditions through the Syslog Trap Mapping Configuration interface.

To launch the Syslog Trap Mapping Configuration interface, run the following command:

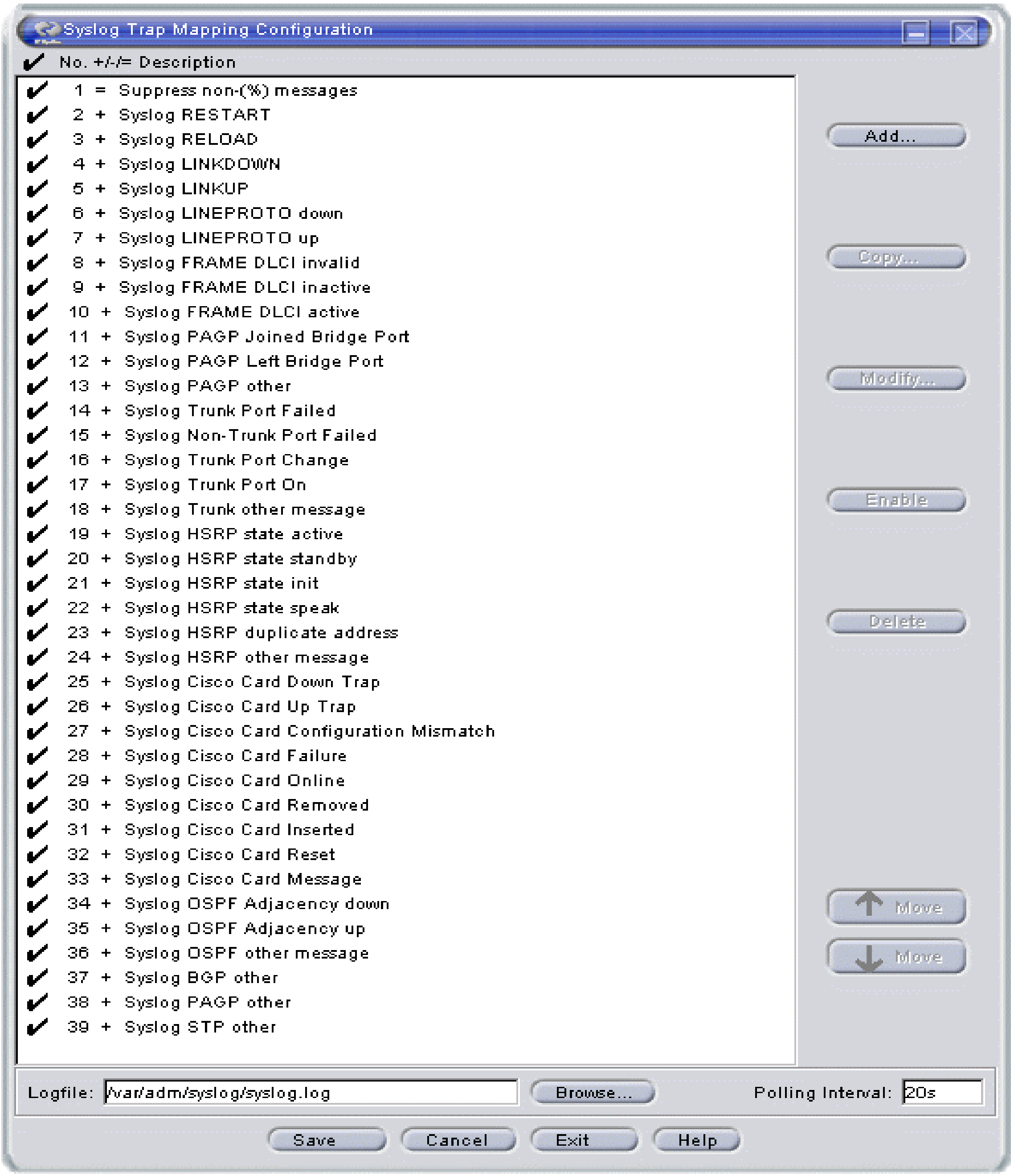
```
$OV_BIN/ovsyslogcfg
```

The main dialog for the Syslog Trap Mapping Configuration interface is shown in Figure 15 on page 249. This dialog supports the following actions:

- Add or delete template conditions.
- Modify template conditions.
- Enable and disable template conditions.
- Reorder template conditions.

For instructions on how to use the Syslog Trap Mapping Configuration interface, see the *Syslog Trap Mapping Configuration* online help.

Figure 15 Syslog Trap Mapping Configuration Dialog



Syslog Trap Mapping Configuration

✓	No.	+/-/=	Description
✓	1	=	Suppress non-(%) messages
✓	2	+	Syslog RESTART
✓	3	+	Syslog RELOAD
✓	4	+	Syslog LINKDOWN
✓	5	+	Syslog LINKUP
✓	6	+	Syslog LINEPROTO down
✓	7	+	Syslog LINEPROTO up
✓	8	+	Syslog FRAME DLCI invalid
✓	9	+	Syslog FRAME DLCI inactive
✓	10	+	Syslog FRAME DLCI active
✓	11	+	Syslog PAGP Joined Bridge Port
✓	12	+	Syslog PAGP Left Bridge Port
✓	13	+	Syslog PAGP other
✓	14	+	Syslog Trunk Port Failed
✓	15	+	Syslog Non-Trunk Port Failed
✓	16	+	Syslog Trunk Port Change
✓	17	+	Syslog Trunk Port On
✓	18	+	Syslog Trunk other message
✓	19	+	Syslog HSRP state active
✓	20	+	Syslog HSRP state standby
✓	21	+	Syslog HSRP state init
✓	22	+	Syslog HSRP state speak
✓	23	+	Syslog HSRP duplicate address
✓	24	+	Syslog HSRP other message
✓	25	+	Syslog Cisco Card Down Trap
✓	26	+	Syslog Cisco Card Up Trap
✓	27	+	Syslog Cisco Card Configuration Mismatch
✓	28	+	Syslog Cisco Card Failure
✓	29	+	Syslog Cisco Card Online
✓	30	+	Syslog Cisco Card Removed
✓	31	+	Syslog Cisco Card Inserted
✓	32	+	Syslog Cisco Card Reset
✓	33	+	Syslog Cisco Card Message
✓	34	+	Syslog OSPF Adjacency down
✓	35	+	Syslog OSPF Adjacency up
✓	36	+	Syslog OSPF other message
✓	37	+	Syslog BGP other
✓	38	+	Syslog PAGP other
✓	39	+	Syslog STP other

Logfile: /var/adm/syslog/syslog.log

Polling Interval: 20s

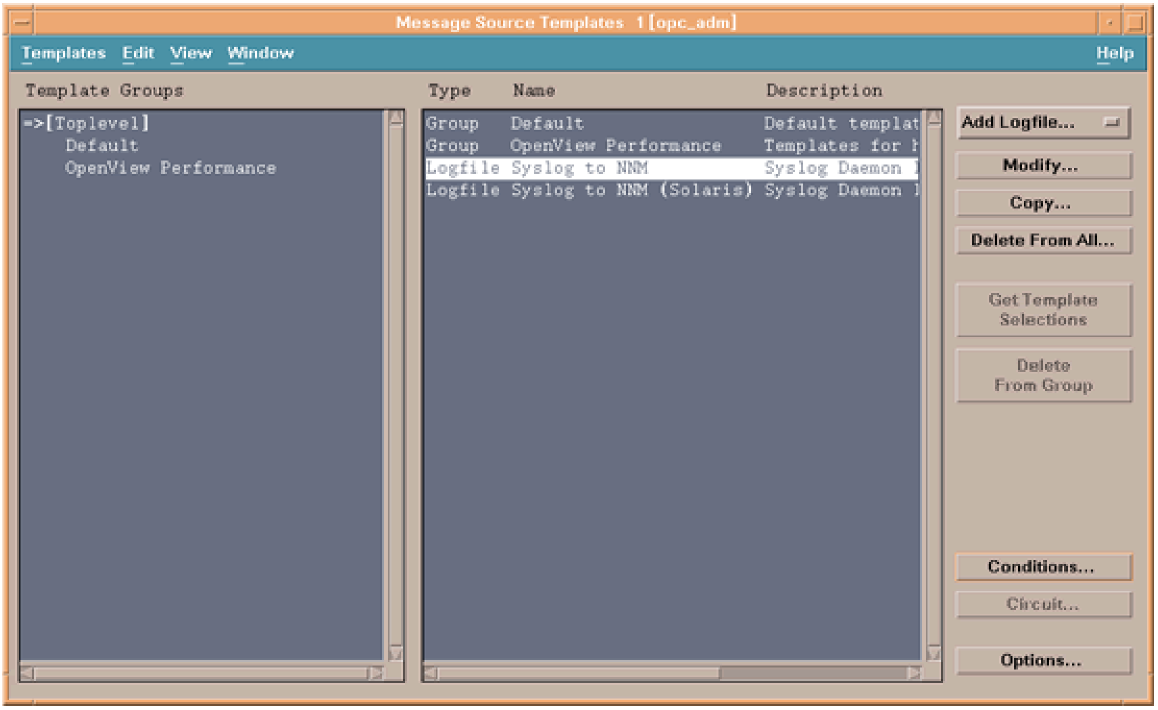
OVO Message Source Templates Window

When the OVO with NNM configuration option is deployed, you construct and modify Syslog Integration message source template conditions through the Message Source Templates configuration window.

To open the Message Source Templates configuration window, launch HP OpenView Operations ($OV_BIN/OpC/opc) as an Administrator or root and then click **Window** → **Message Source Templates**.

The Message Source Templates window is shown in Figure 16. The Syslog to NNM template for HP-UX operating systems and Syslog to NNM (Solaris) template for Solaris operating systems contain the conditions that map syslog message to OpenView SNMP traps.

Figure 16 OVO Message Source Templates Window



Template Groups

```
=>[Toplevel]
    Default
    OpenView Performance
```

Type	Name	Description
Group	Default	Default templat
Group	OpenView Performance	Templates for H
Logfile	Syslog to NNM	Syslog Daemon I
Logfile	Syslog to NNM (Solaris)	Syslog Daemon I

With the NNM to Syslog template for your operating system selected, you can use the following actions to modify properties and conditions:

- To modify the properties of the template, select **Modify**.

- To modify the template conditions, select **Conditions**.

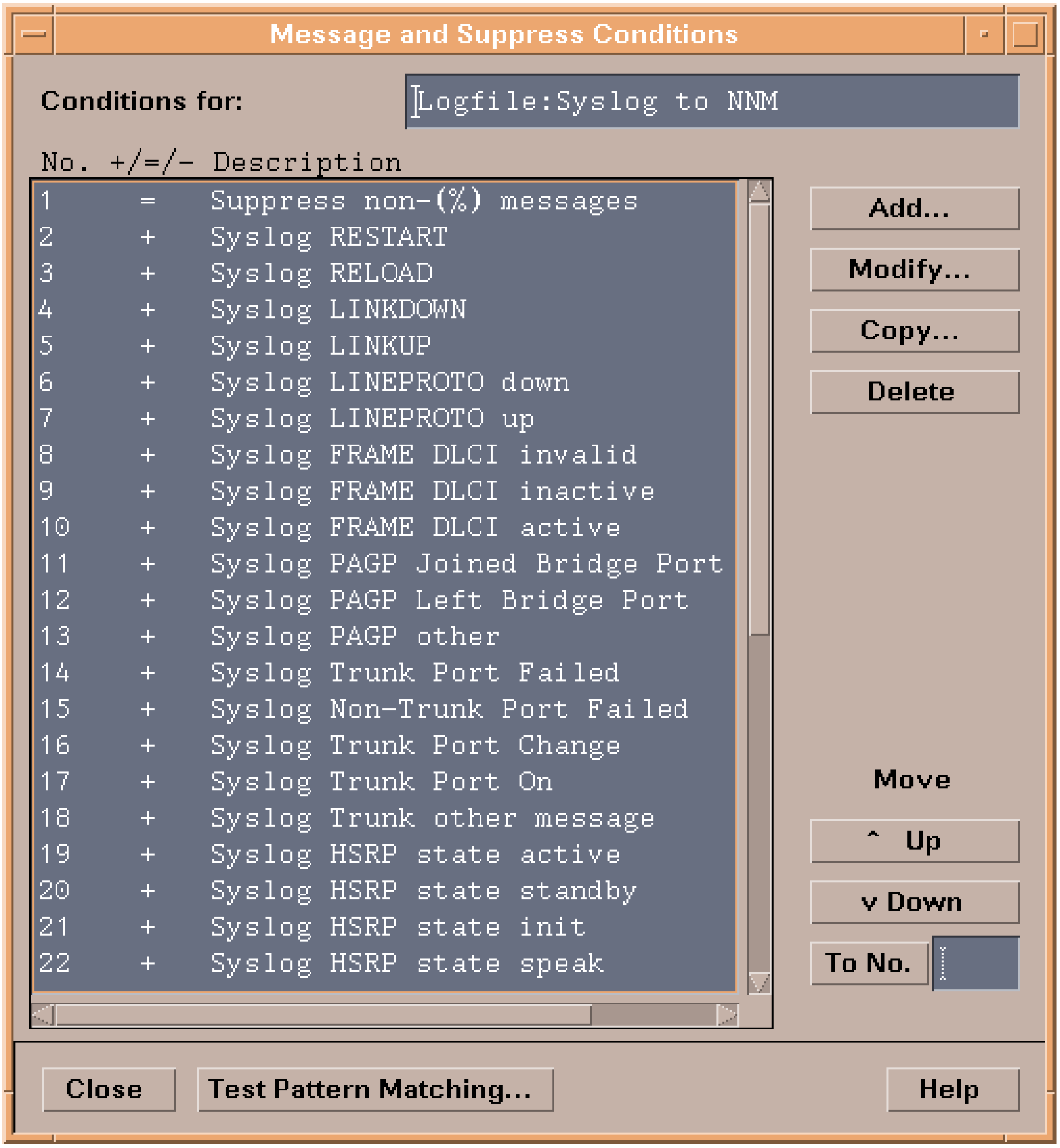
For instructions on how to use the Message Source Templates window, see the *OVO Administrator* online help.

Understanding the Syslog to NNM Template

NNM includes out-of-the-box template conditions for which syslog messages are mapped to OpenView SNMP traps. Each type of syslog message to be mapped is defined in one template condition. The conditions are contained in the Syslog to NNM template.

The Syslog to NNM template contains many out-of-the-box conditions. Figure 17 on page 252 shows the template conditions as they appear in the OVO Message and Suppress Conditions window.

Figure 17 Syslog to NNM Template Conditions



Conditions for: Logfile:Syslog to NNM

```
No. +/=/- Description
1      =    Suppress non-(%) messages
2      +    Syslog RESTART
3      +    Syslog RELOAD
4      +    Syslog LINKDOWN
5      +    Syslog LINKUP
6      +    Syslog LINEPROTO down
7      +    Syslog LINEPROTO up
8      +    Syslog FRAME DLCI invalid
9      +    Syslog FRAME DLCI inactive
10     +    Syslog FRAME DLCI active
11     +    Syslog PAGP Joined Bridge Port
12     +    Syslog PAGP Left Bridge Port
13     +    Syslog PAGP other
14     +    Syslog Trunk Port Failed
15     +    Syslog Non-Trunk Port Failed
16     +    Syslog Trunk Port Change
17     +    Syslog Trunk Port On
18     +    Syslog Trunk other message
19     +    Syslog HSRP state active
20     +    Syslog HSRP state standby
21     +    Syslog HSRP state init
22     +    Syslog HSRP state speak
```

In both the OVO template editor windows and the NNM Syslog Trap Mapping Configuration interface, the order of the conditions is important for pattern matching. The patterns are tested in the order that they are listed, and the first pattern to match is executed.

Since ordering of template conditions matters, it is important to place suppression patterns and more specific patterns at the beginning of the list. More general patterns should go last.

The first condition is a suppress unmatched condition, meaning the pattern will exclude any message that does not conform to its pattern. In this case, the pattern matches only those syslog messages with the % character as the leading non-white space character in the message (specifically, Cisco syslog message types). This pattern does not functionally perform anything, but is significant as an optimization tool, since all other conditions in this template will execute only on Cisco syslog messages.

The remaining conditions look for Cisco syslog messages matching defined patterns as identified in Table 17.

Table 17 Template Conditions and Corresponding Syslog Messages

Template Condition Name	Syslog Message Format
Syslog LINEPROTO down	`%LINEPROTO-5-UPDOWN (down)`
Syslog LINEPROTO up	`%LINEPROTO-5-UPDOWN (up)`
Syslog FRAME DLCI Invalid	`%FR-5-DLCICHANGE (INVALID)`
Syslog FRAME DLCI Inactive	`%FR-5-DLCICHANGE (INACTIVE)`
Syslog FRAME DLCI Active	`%FR-5-DLCICHANGE (ACTIVE)`
Syslog OSPF Adjacency up	`%OSPF-5-ADJCHG (FULL)`
Syslog OSPF Adjacency down	`%OSPF-5-ADJCHG (DOWN)`
Syslog LINKUP	`%LINK-3-UPDOWN (up)`
Syslog LINKDOWN	`%LINK-3-UPDOWN (down)`
Syslog HSRP state speak	`%STANDBY-6-STATECHANGE (Speak)`
Syslog HSRP state standby	`%STANDBY-6-STATECHANGE (Standby)`

Table 17 Template Conditions and Corresponding Syslog Messages

Template Condition Name	Syslog Message Format
Syslog HSRP state active	`%STANDBY-6-STATECHANGE (Active)`
Syslog HSRP state init	`%STANDBY-6-STATECHANGE (Init)`
Syslog HSRP duplicate address	`%STANDBY-3-DUPADDR`
Syslog Cisco Card Up(Down) Trap	`%SNMP-5-MODULETRAP`

Syslog Cisco Card Failure	`%SYS-5-MOD_NORESPONSE`
	`%SYS-3-MOD_FAIL`
	`%SYS-3-MOD_FAILREASON`

Syslog Cisco Card Online	`%SYS-5-MOD_OK`
Syslog Cisco Card Removed	`%SYS-5-MOD_REMOVE`
Syslog Cisco Card Inserted	`%SYS-5-MOD_INSERT`
Syslog Cisco Card Reset	`%SYS-5-MOD_RESET`
Syslog Cisco Card Configuration Mismatch	`%SYS-3-MOD_CFGMISMATCH[1,2,3,4]`
Syslog PAGP Joined Bridge Port	`%PAGP-5-PORTTOSPT`
Syslog PAGP Left Bridge Port	`%PAGP-5-PORTFROMSPT`
Syslog Trunk Port Failed	`%DTP-3-TRUNKPORTFAIL`
Syslog Non-Trunk Port Failed	`%DTP-3-NONTRUNKPORTFAIL`
Syslog Trunk Port On	`%DTP-5-TRUNKPORTON`
Syslog Trunk Port Change	`%DTP-5-TRUNKPORTCHG`
Syslog OSPF other message	`%OSPF`
Syslog HSRP other message	`%STANDBY`
Syslog PAGP other	`%PAGP`
Syslog Cisco Card Message	`%SYS-`*n*`-MOD`

Table 17 Template Conditions and Corresponding Syslog Messages

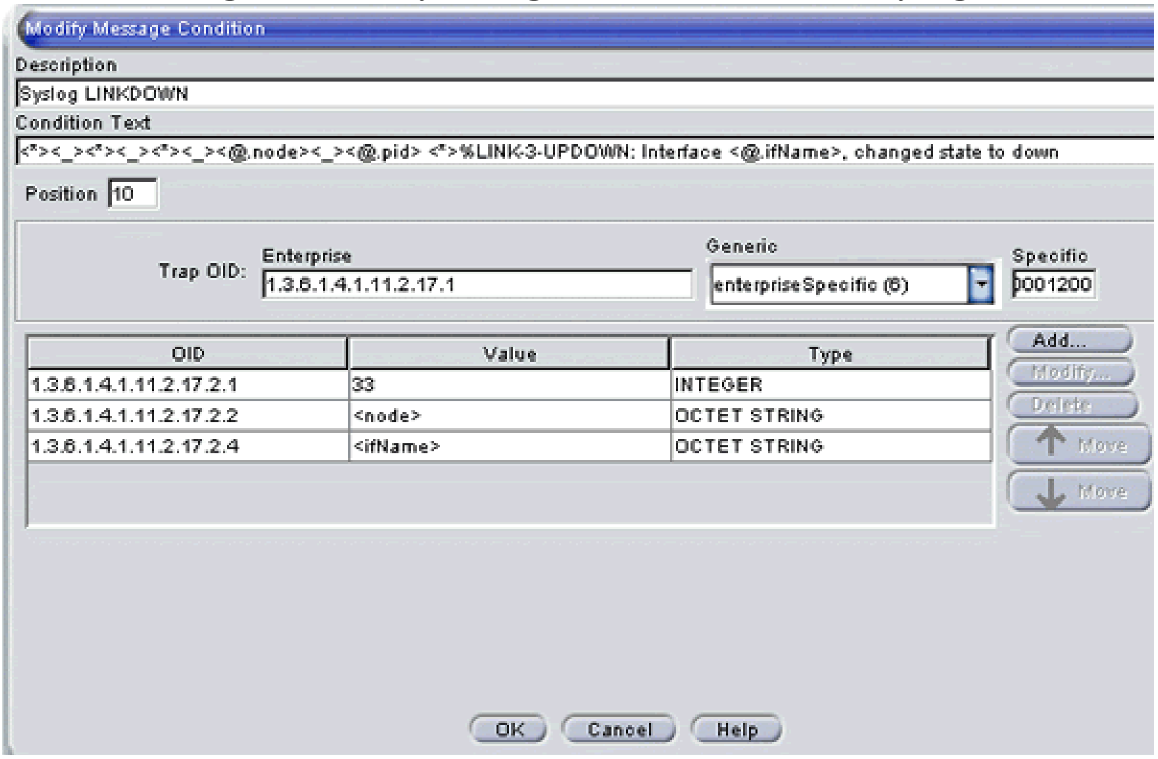
Template Condition Name	Syslog Message Format
Syslog Trunk other	`%DTP`
Syslog STP other	`%SPANTREE`
Syslog CDP	`%CDP`
Syslog BGP other message	`%BGP`

How Syslog to NNM Conditions Function in the NNM Standalone Configuration

Figure 18 on page 256 shows a condition window to modify a Syslog to NNM template condition definition. The window is divided into two logical parts: a condition text field to match patterns and a set of attributes that define the SNMP trap to be generated.

The Condition Text field uses syntax similar to a regular expression. It identifies the pattern to match and names any subexpressions in the pattern to be used in the mapping to an SNMP trap. See the *NNM Syslog Trap Mapping Configuration* online help for more information about pattern matching.

Figure 18 Modify Message Condition Window for Syslog LINKDOWN



Description: Syslog LINKDOWN

Condition Text: <*><_><*><_><*><_><@.node><_><@.pid> <*>%LINK-3-UPDOWN: Interface <@.ifName>, changed state to down

Position: 10

Trap OID: Enterprise: 1.3.6.1.4.1.11.2.17.1 Generic: enterpriseSpecific (6) Specific: 0001200

OID	Value	Type
1.3.6.1.4.1.11.2.17.2.1	33	INTEGER
1.3.6.1.4.1.11.2.17.2.2	<node>	OCTET STRING
1.3.6.1.4.1.11.2.17.2.4	<ifName>	OCTET STRING

The Position field identifies the location of the condition with respect to the other conditions of the template.

The Trap OID is defined by the enterprise, generic, and specific fields. The trap OID is used to determine the type of OpenView event to be generated in response to a message matching the condition pattern.

The varbinds of the trap are defined in the lower table, as shown in Figure 18.

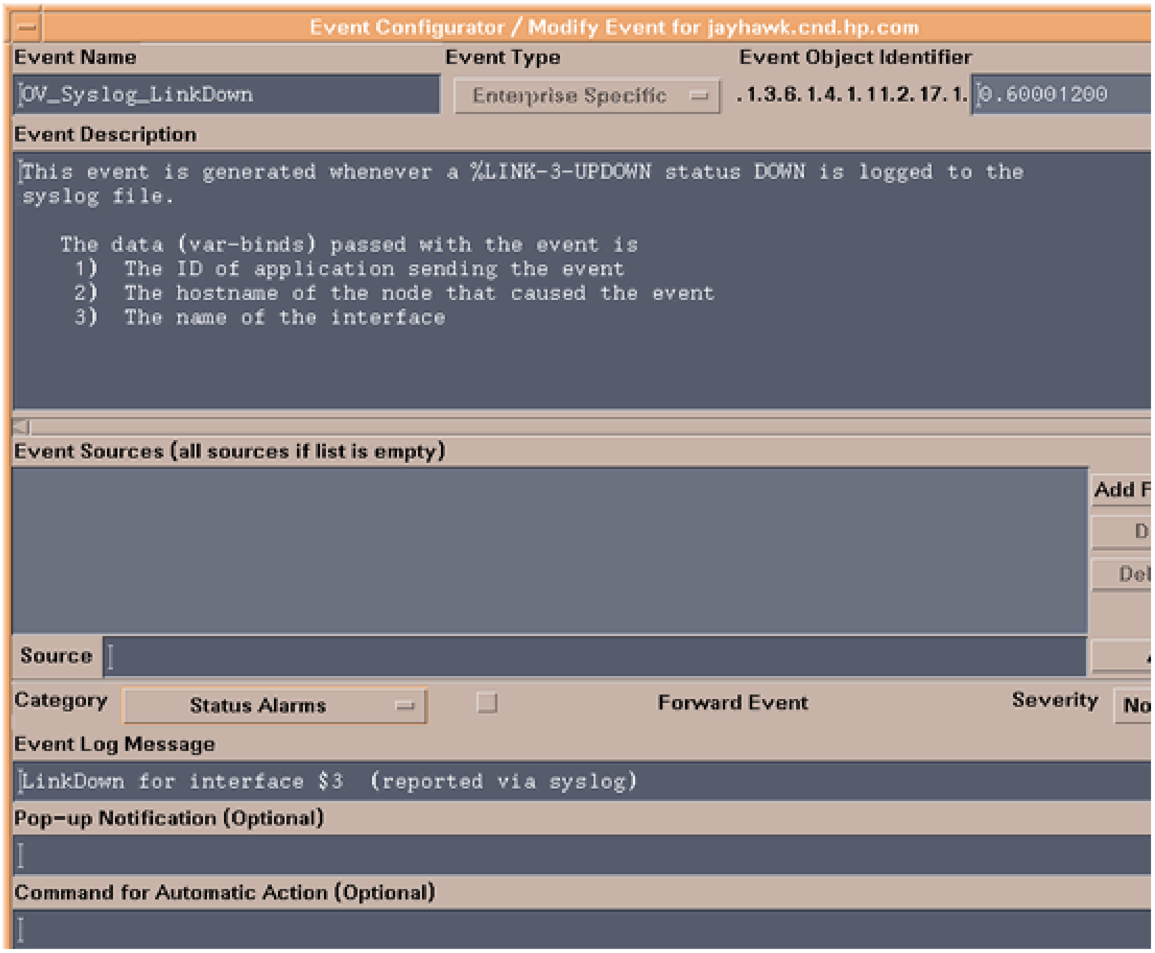
You can edit the Condition Text and the Trap OID fields. You can also modify and reorder any of the varbinds. See the *NNM Syslog Trap Mapping Configuration* online help for more information about modifying these fields.

Messages matching the pattern defined in the Condition Text field cause the OpenView event identified by the Trap OID to be generated. For example, when a %LINK-3-UPDOWN status DOWN message is logged to the syslog file, the message is intercepted because the Syslog LINKDOWN condition looks for this pattern (as shown in the Condition Text field of Figure 18). In that same condition, a trap OID is identified, which corresponds to the OpenView event, OV_Syslog_LinkDown, as shown in Figure 19 on page 258. This event is then generated.

To view or identify the corresponding OpenView event to be generated, do the following:

1. Start the NNM UI by typing: ovw
2. From the Root window, click **Options** → **Event Configuration**.
3. Select **OpenView** from the Enterprise Name list. A list of OpenView events displays in the bottom pane.
4. Locate the trap OID from the Event Identifier list.
5. Double-click the event or click **Edit** → **Modify Event** to display the Event Configurator/Modify Event window. An example is shown in Figure 19 on page 258.

Figure 19 OV_Syslog _LinkDown Event Configuration Window



Event Configurator / Modify Event for jayhawk.cnd.hp.com

Event Name: OV_Syslog_LinkDown

Event Type: Enterprise Specific

Event Object Identifier: .1.3.6.1.4.1.11.2.17.1. 0.60001200

Event Description

```
This event is generated whenever a %LINK-3-UPDOWN status DOWN is logged to the
syslog file.

   The data (var-binds) passed with the event is
    1)  The ID of application sending the event
    2)  The hostname of the node that caused the event
    3)  The name of the interface
```

Event Sources (all sources if list is empty)

Source:

Category: Status Alarms

Forward Event

Severity: No

Event Log Message

```
LinkDown for interface $3  (reported via syslog)
```

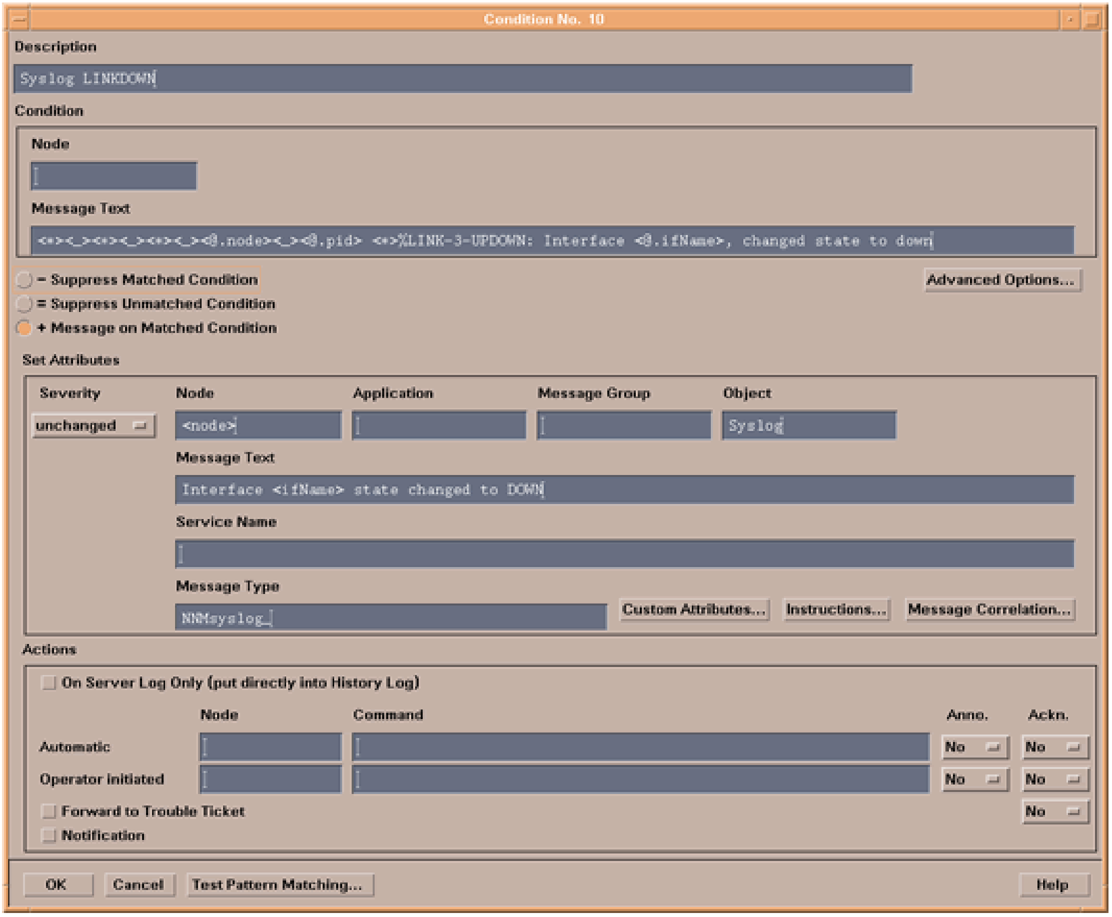
Pop-up Notification (Optional)

Command for Automatic Action (Optional)

How Syslog to NNM Templates Function in the OVO with NNM Configuration

Figure 20 on page 259 shows a condition window to modify a Syslog to NNM template condition definition. The window is divided into three logical parts: input section, condition matching section, and message output section.

Figure 20 OVO Condition Editing Window



Condition No. 10

Description: Syslog LINKDOWN

Condition

Node:

Message Text: <*><_><*><_><*><_><@.node><_><@.pid> <*>%LINK-3-UPDOWN: Interface <@.ifName>, changed state to down

- = Suppress Matched Condition
- = Suppress Unmatched Condition
- [✓ + Message on Matched Condition]

Advanced Options...

Set Attributes

Severity	Node	Application	Message Group	Object
unchanged	<node>			Syslog

Message Text: Interface <ifName> state changed to DOWN

Service Name:

Message Type: NNMsyslog

Custom Attributes... Instructions... Message Correlation...

Actions

On Server Log Only (put directly into History Log)

	Node	Command	Anno.	Ackn.
Automatic			No	No
Operator initiated			No	No
Forward to Trouble Ticket				No
Notification				

OK Cancel Test Pattern Matching... Help

The top part contains the input section. Messages are matched according to values stored in the fields listed in Table 18.

Table 18 Input Fields for OVO Template Conditions

Field	Description	Size
Node	Course-grained identifier for the source of a message, such as software.hp.com.	254

Table 18 Input Fields for OVO Template Conditions

Field	Description	Size
Message Text	Content and/or description of a message. Use OVO's regular expression-like syntax to define the Message Text. Right-click in the field to view a short list of acceptable regular expressions. For example, if you want to match messages on the string "`Switch1`", then define the Message Text field to be `<*>Switch1<*>`. See the *OVO Administrator* online help for more information on writing pattern matching expressions.	512

The matching conditions section describes how the conditions are to be treated. The options are listed in Table 19.

Table 19

Option	Description
Suppress Matched Condition	Suppresses all messages matching condition fields in the input section. Messages are stored in a log file and are not displayed in the OVO message browser. This is identified by the – symbol.
Suppress Unmatched Condition	Suppresses all messages not matching condition fields in the input section. Messages are stored in a log file and are not displayed in the OVO message browser. This is identified by the = symbol.
Message on Matched Condition	Forwards all messages matching condition fields in the input section to the OVO message browser. This is identified by the + symbol.

The lower portion contains the output section. The fields in this section correspond to columns in the message browser. Use these fields to reformat the original message into a more readable format for end users. When any of these fields are unspecified, values are copied from the original message. Table 20 lists the key message fields, their intended usage, and their size limitations.

If the value of an output field is a named subexpression from the Message Text input field, it must be enclosed in angle brackets (<>).

Table 20 Output Fields of OVO Template Conditions

Field	Description	Size
Node	Identifies the source of a message. In the `Syslog to NNM` template conditions, the value of this field is pulled from the incoming syslog message. Its value is stored in the variable node.	254
Application	Medium-grained identifier for a message source; for example, Oracle.	32
Message Group	Group of alarms to which a message belongs.	32
Object	Fine-grained message source identifier; for example, Syslog.	32
Message Text	Contains the description text of a message.	2048
Service Name	Identifier used to associate a message with a service.	254
Message Type	Identifier of a subgroup of a message group. In order for syslog messages to be forwarded to the NNM `syslogTrap` process, `NNMsyslog_` must be entered in this field.	32

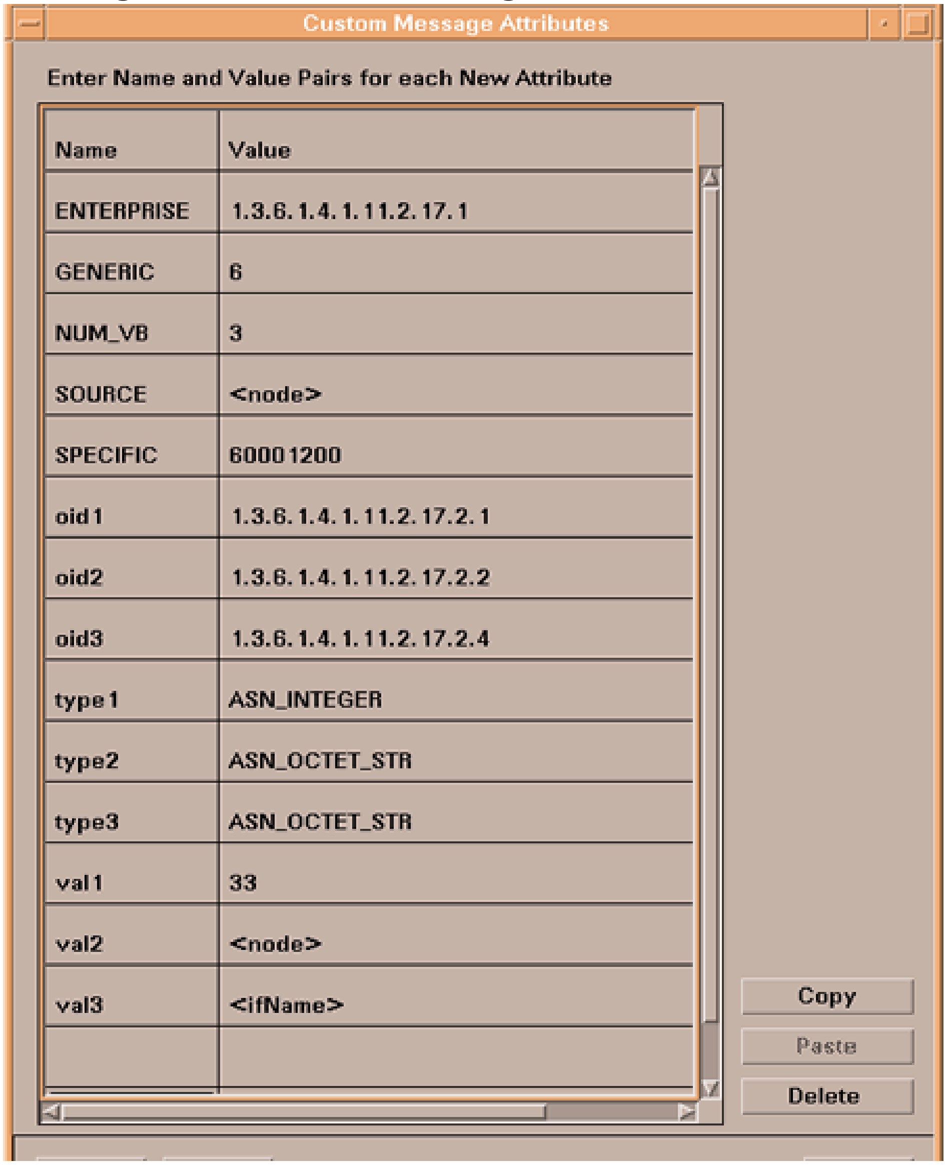
Be aware that Node, Application, Message Group, and Object fields are used by administrators to create filtered views in the OVO interface. Consistent use of these fields is paramount to enabling operators to effectively monitor equipment for which they are responsible.

From the OVO condition editing window, as shown in Figure 20 on page 259, you add, delete, or modify custom message attributes (CMAs). Custom message attributes allow you to add your own attributes to a message. This means that in addition to the default message attributes, you can extend OVO with attributes of your choice.

To assign custom message attributes to a message, use the Custom Message Attributes window. To open the window, select **Custom Attributes** from the OVO condition editing window. For example, Figure 21 on page 264 shows a list of defined custom message attributes for the syslog LINKDOWN condition.

The Name field defines the name of the attribute that is displayed as an additional column in the message browser. The Value field sets the value of the attribute. A Value entry can contain either hard-coded text or a variable defined in the Message Text input field.

Figure 21 OVO Custom Message Attributes



Enter Name and Value Pairs for each New Attribute

Name	Value
ENTERPRISE	1.3.6.1.4.1.11.2.17.1
GENERIC	6
NUM_VB	3
SOURCE	<node>
SPECIFIC	60001200
oid1	1.3.6.1.4.1.11.2.17.2.1
oid2	1.3.6.1.4.1.11.2.17.2.2
oid3	1.3.6.1.4.1.11.2.17.2.4
type1	ASN_INTEGER
type2	ASN_OCTET_STR
type3	ASN_OCTET_STR
val1	33
val2	<node>
val3	<ifName>

If you have enabled the display of custom message attributes in your OVO message browser, they appear as additional columns in the browser.

Maintaining Syslog Integration

This section provides instructions for tasks that an NNM administrator should perform to maintain the working of the Syslog Integration functionality.

Administrative Tasks for NNM Standalone Configurations

Deploying Syslog to NNM Template

After you edit syslog message source template conditions with the Syslog Trap Mapping Configuration interface, run the following command to deploy the new configuration:

```
$OV_BIN/setupSyslog.ovpl -standalone -deploy
```

The -deploy command option generates and encrypts the new template and restarts the embedded OVO agent so that the new template is reloaded.

Testing Patterns in Template Conditions

Before you redeploy the Syslog to NNM template, you can verify the syntax of any template condition with the following command:

```
$OV_BIN/OpC/utils/opcpat
```

Review the reference page for *opcpat* for instructions on how to use the command. Details about how to access reference pages are in the online help.

Disabling Syslog Integration Functionality

To disable the syslog functionality, run the following command:

```
$OV_BIN/setupSyslog.ovpl -standalone -disable
```

This command stops the embedded OVO agent processes and the NNM syslogTrap process. The OVO agent software remains on the NNM management station. To remove the OVO agent software, see Removing Syslog Integration on page 245.

You can re-enable the Syslog Integration functionality with the following command:

```
$OV_BIN/setupSyslog.ovpl -standalone
```

Administrative Tasks for OVO with NNM Configurations

Disabling Syslog Integration Functionality

To disable the Syslog Integration functionality, run the following command:

```
$OV_BIN/setupSyslog.ovpl -server -disable
```

Starting and Stopping syslogTrap

To start the NNM syslogTrap process, run the following command:

```
ovstart -c syslogTrap
```

This process is not registered with NNM until you execute the `setupSyslog.ovpl` configuration script. Therefore, you cannot start this process until you have run the `setupSyslog.ovpl` script.

To stop the syslogTrap process, run the following command:

```
ovstop -c syslogTrap
```

Troubleshooting Tips

This section contains troubleshooting tips for the Syslog Integration functionality.

Performance

The Syslog Integration functionality is not intended for high volume syslog message systems.

Performance issues may arise if the syslog messages from managed network elements become abundant. You may need to tune the Syslog to NNM template conditions to improve performance. Additionally, you may need to add a filtering mechanism to the NNM background process (syslogTrap) that maps the syslog message to an SNMP trap.

Configuration

Duplicate Syslog Messages in Message Browser:

If you see duplicate syslog messages in your message browser, consider one of the following possible reasons:

- In OVO with NNM configurations, you must enable the message stream interface for both the Syslog to NNM template and the OVO agent on the NNM management station in order for syslog messages to be processed. However, in OVO, there are a multitude of combinations for diverting messages through the system. For example, you can enable the message stream interface for individual conditions of a template to copy messages. You can also enable the message stream interface for the template to copy messages. This will produce multiple messages in the message browser.
- If templates are not correctly ordered, a message may match conditions of multiple templates. When in occurs, multiple messages are displayed in the message browser. For example, if a wildcard template is assigned and installed on a system, then every message entering the agent is forwarded to the message browser. Furthermore, if additional templates are assigned and installed on a system, then those messages matching the conditions of the templates are also forwarded to the message browser.

No Syslog Messages in Message Browser

If you do not see syslog messages in your message browser, consider one of the following possible reasons:

- The Syslog to NNM template is not installed or enabled on the OVO agent system (on the NNM management station). To verify that the Syslog to NNM template is installed and enabled on the OVO agent, run the following command:

  ```
  $OV_BIN/OpC/opctemplate
  ```

  This command lists all templates with the type, name, and status. This command is helpful to determine if a template you have assigned to an agent node has successfully been installed on that agent system.

  Be aware that this command does not indicate which version of the template has been deployed. If you have made modifications to any assigned templates, you must reinstall the templates on the managed nodes.

- For OVO with NNM configurations, the message stream interface is not enabled for either the Syslog to NNM template or the OVO agent on the NNM management station.

  To isolate the problem, you can turn on XPL tracing for the syslogTrap process. If you see no activity in incoming messages, it typically means that the message stream interface has not been enabled in all places where it must be enabled.

- The templates are marked as log only. Use the NNM Event Configuration window to change the logging state. From any NNM submap, select **Options** → **Event Configuration**.

- When APA is enabled, several correlators are added to HP OpenView Correlation Composer that listen for and correlate syslog messages. You may not see some syslog messages in the Alarm Browser because the APA-enabled correlators may discard the syslog messages or nest them under APA status alarms. For a description of the APA correlators and how syslog messages take part in event reduction, see APA and Event Reduction on page 140.

8 Advanced Routing Smart Plug-In

What the Advanced Routing Smart Plug-in Discovers

The Advanced Routing Smart Plug-in (SPI) enables Extended Topology to discover information about Hot Standby Routing Protocol (HSRP) and Virtual Redundancy Routing Protocol (VRRP) groups, Open Shortest Path First (OSPF) areas, and devices that support IPv6. You must purchase and install a license for the Advanced Routing SPI to use this functionality. To review additional network NNM SPIs, point your browser to `http://openview.hp.com` and follow the links to OpenView products.

Discovering HSRP Information

The Advanced Routing SPI enables Extended Topology to discover and display HSRP information from managed devices that support the HSRP protocol.

While HSRP discovery is automatic, there are important preliminary steps you need to take to ensure correct HSRP discovery and monitoring. If you want to discover and monitor HSRP groups, see Running HSRP Discovery on page 272 for details.

The Active Problem Analyzer polls HSRP routers and reports HSRP group status information. See Understanding HSRP View Status Information on page 276 for more information.

After HSRP discovery and monitoring is functional, see Using the HSRP View to Diagnose Network Problems on page 278 or the online help for information on using this feature.

Discovering VRRP Information

The Advanced Routing SPI enables Extended Topology to discover and display VRRP information from managed devices that support the VRRP protocol.

While VRRP discovery is automatic, there are important preliminary steps you need to take to ensure correct VRRP discovery and monitoring. If you want to discover and monitor VRRP groups, see Running VRRP Discovery on page 279 for details.

The Active Problem Analyzer polls VRRP routers and reports VRRP group status information. See Understanding VRRP View Status Information on page 283 for more information.

After VRRP discovery and monitoring is functional, see Using the VRRP View to Diagnose Network Problems on page 285 or the online help for information on using this feature.

Discovering OSPF Information

The Advanced Routing SPI enables Extended Topology to discover and display OSPF information. OSPF discovery is a separate process from the Extended Topology discovery. You run OSPF discovery by using a manual discovery procedure. See Running OSPF Discovery on page 286 for more information about OSPF discovery.

Another product, the HP OpenView Route Analytics Management System (RAMS), together with the NNM/RAMS Integration Module, represents a superior solution to OSPF management. By actively participating in the network protocols, RAMS provides near real-time routing data across multiple protocols. Coupled with NNM Advanced Edition, it greatly enhances the root-cause analysis and visualization of your OSPF routing fabric.

Discovering IPv6 Information

The Advanced Routing SPI enables Extended Topology to discover and display IPv6 information. When you run IPv6 discovery, Extended Topology discovers global, site-local, and link-local addresses. The management station and all routers must be dual-stacked for IPv6 discovery to function properly.

To prepare the Advanced Routing SPI for IPv6 discovery, you must run the following script:

- *UNIX*:

```
$OV_BIN/setupExtTopo.ovpl
```

- *Windows*:

```
%OV_BIN%\setupExtTopo.ovpl
```

See Running IPv6 Discovery on page 292 for more information.

Running HSRP Discovery

HSRP permits two or more routers (in an HSRP Group) to act as a single virtual router. Routers in an HSRP Group share a virtual IP address and a virtual MAC address.

Each router has a corresponding fixed IP address on the interface that is participating in the HSRP Group.

For the HSRP feature to work correctly, the following conditions must be true:

- NNM must *not* attempt to discover and manage the virtual IP address of an HSRP Group.
- NNM *must* manage the fixed IP address of router interfaces in the HSRP Group.

NNM meets both of those conditions, internally tracking the virtual IP address of an HSRP group while managing the fixed IP address of router interfaces in the HSRP Group.

Checking NNM for Correct Handling of HSRP Virtual IP Addresses

To make sure NNM is configured for correctly handling HSRP virtual addresses, use the following procedure:

1. As `Administrator` or `root`, edit the following file:
   - *UNIX*:

     ```
     $OV_LRF/netmon.lrf
     ```
   - *Windows*:

     ```
     %OV_LRF%\netmon.lrf
     ```
2. Look for the following text within the file and verify that the bold text is present.

   `OVs_YES_START:ovtopmd,pmd,ovwdb:-P -k segRedux=true` **`-k migrateHsrpVirtualIP=true`** `:OVs_WELL_BEHAVED:15:PAUSE`
3. If the bold text is present, then NNM is handling the HSRP virtual IP address correctly.

4. If the bold text is not present add the text and save the file. You also need to restart the `netmon` process as follows. As `Administrator` or `root`, do the following:
   a. Run the `ovstop netmon` command.
   b. Run the `ovaddobj netmon.lrf` command.
   c. Run the `ovstart netmon` command.

HSRP Discovery with Pre-existing NNM Topology

If you already have an NNM topology database (for example, if you have installed NNM and proceeded with an automatic discovery of your network), and the contents of the `netmon.lrf` shows `migrateHsrpVirtualIP=false` or does not contain the `migrateHsrpVirtualIP parameter,` then the database is likely to contain the virtual IP addresses of your HSRP groups. This is especially true if you used the auto-discovery feature.

You have two tasks:

1. Verify that NNM is correctly managing HSRP virtual IP addresses in your environment.
2. Remove from NNM any HSRP virtual IP addresses that NNM currently manages.

The next sections explain the procedure you should follow.

Verifying that NNM is Properly Managing HSRP Virtual IP Addresses

Follow the procedure shown in Checking NNM for Correct Handling of HSRP Virtual IP Addresses on page 272.

Remove Virtual IP Addresses from the NNM Topology

Next you need to remove any virtual IP addresses from the existing NNM topology data. There are a couple of common ways to do this:

- Use the following command to completely delete the router from the database (deleted routers get re-discovered in the next step, but without their virtual interfaces):

  — *UNIX*:

  ```
  $OV_BIN/ovtopofix -r <router>
  ```

  — *Windows*:

  ```
  %OV_BIN%\ovtopofix -r <router>
  ```

  You should remove the *node* (using the NNM ID) rather than the IP address. The NNM ID can be obtained with the following command:

  — *UNIX*:

  ```
  $OV_BIN/ovtopodump <router>
  ```

  — *Windows*:

  ```
  %OV_BIN%\ovtopodump <router>
  ```

  See the *ovtopofix(1M)* and *ovtopodump(1M)* reference pages for more information. Details about how to access reference pages are in the online help.

- Alternatively, from the NNM graphical interface (`ovw`), find each of your HSRP routers. Open each one to show all its interfaces. Delete the interface that corresponds to the virtual IP address.

After you have removed the virtual IP addresses from the NNM topology data, you need to rediscover the router.

Validating Your Results

To verify that your HSRP discovery is correct, restart Extended Topology discovery. See Start Extended Topology Discovery on page 23 for more information.

When the discovery finishes, open the HSRP Group Detail screen for each group and check for the following:

- If the HSRP virtual IP address shows up in both the IP Interface column and the Group Membership column, NNM is most likely managing the virtual IP address. You need to remove this interface from NNM (see Remove Virtual IP Addresses from the NNM Topology on page 274).
- You may see an HSRP Group with only one router, even though an HSRP Group must have at least two routers. This could have several causes. To resolve the problem, do the following:
    - Ensure that the actual IP address of the missing HSRP router is in NNM by running the following command:
        - *UNIX*:

          ```
          $OV_BIN/ovtopodump <fixed IP address>
          ```

        - *Windows*:

          ```
          %OV_BIN%\ovtopodump <fixed IP address>
          ```

    - Make sure that NNM has SNMP access to the missing router
    - Rerun discovery when all the HSRP router interfaces are up

      Note that if an HSRP router interface is down during discovery, the Advanced Routing SPI will discover HSRP groups incompletely. This is because the HSRP MIBs at the time of discovery reflect the actual state. This should be a transient problem that resolves when an Extended Topology discovery occurs after the interface is back up.

Understanding HSRP View Status Information

The Active Problem Analyzer polls HSRP routers and reports HSRP group status information. See Table 21 for a list of the various group status colors and their meanings.

Table 21 HSRP Group Status

HSRP Group Status Color	HSRP Group Status Description
Unknown (blue)	The HSRP group has not been polled since discovery completed, so the state of the group has not been determined yet. The actual state of this HSRP group will be displayed as soon as it is polled.
Normal (green)	The HSRP group is correctly configured and operational. This HSRP group is providing routing functionality, and all standby routers are available.
Warning (cyan)	The HSRP group has an interface in the `Active` state, and an interface in the `Standby` state, but it also has at least one interface that is not in the `Listen` state. This HSRP group is providing routing functionality. However, one or more standby routers are definitely unavailable.
Minor or Marginal (yellow)	The HSRP Group has an interface in the `Active` state, but there is no interface in the HSRP group which is in the `Standby` state. This HSRP group is providing routing functionality, but *there is no standby router available*.
Major (orange)	Multiple interfaces in the HSRP group are in the `Active` state, or multiple interfaces in the HSRP group are in the `Standby` state. This HSRP group is *not functioning correctly*. There is almost certainly a problem with routing functionality.

Table 21 HSRP Group Status

HSRP Group Status Color	HSRP Group Status Description
Critical (red)	The HSRP group has no interface in the `Active` state. This HSRP group is *definitely not* providing routing functionality.

The Active Problem Analyzer displays the following alarms in the alarms browser.

- `OV_HSRP_No_Active`: The reported HSRP group is completely inoperational.
- `OV_HSRP_Multiple_Active`: The reported HSRP group is in an abnormal condition as there are multiple active interfaces.
- `OV_HSRP_No_Standby`: The reported HSRP group has no standby interface.
- `OV_HSRP_Degraded`: The reported HSRP group has an interface that is not functioning properly
- `OV_HSRP_FailOver`: The reported HSRP group has changed its active interface.
- `OV_HSRP_Standby_Changed`: The reported HSRP group has changed its standby interface.
- `OV_HSRP_Normal`: The reported HSRP group is now functioning normally.

See the *trapd.conf* reference page for more information. Details about how to access reference pages are in the online help.

Potential Status Problem

The following condition can lead the Advanced Routing SPI to yield unexpected status for HSRP groups.

- If you run an Extended Topology discovery while any HSRP group interfaces are down, Extended Topology will not discover all of the HSRP group interfaces and will calculate and display HSRP group status using incomplete information. After you run an Extended Topology discovery that finds all of the HSRP group's router interfaces, Extended Topology will show accurate HSRP status.

⚠ As you add equipment with new HSRP virtual IP addresses to your managed environment, you need to add these addresses to the `netmon.noDiscover` file prior to turning on the new equipment. See the reference page for *netmon.noDiscover* for usage information. Details about how to access reference pages are in the online help. If you fail to do this, and NNM discovers these virtual IP addresses, you will need to complete the tasks explained in HSRP Discovery with Pre-existing NNM Topology on page 273.

Using the HSRP View to Diagnose Network Problems

The following scenario gives one example of using Extended Topology information to diagnose a network problem with HSRP. This discussion assumes you are familiar with HSRP.

Suppose you have two directly connected router interfaces, interface a on router 1 and interface b on router 2. This discussion refers to these router interfaces as interfaces 1.a and 2.b respectively.

Assume both interfaces are located within the same LAN segment. You have router interface 1.a configured as `active` and router interface 2.b configured as `standby` for standby group a.b.c.d. a.b.c.d represents the virtual IP address of the standby group.

In this example, interface 1.a fails, and the following situation occurs:

1. Interface 2.b becomes active and generates an `OV_HSRP_FailOver` alarm.
2. To investigate this alarm, you open an HSRP Group view by selecting the `OV_HSRP_Failover` alarm in the Alarm Browser and use the `Actions:Views-> HSRP Group Detail` menu.
3. The `HSRP Group Detail` view shows you that interface 2.b is now active and interface 1.a is down. You need to troubleshoot interface 1.a.

Running VRRP Discovery

VRRP permits two or more routers (in a VRRP Group) to act as a single virtual router. Routers in a VRRP Group share a virtual IP address and a virtual MAC address.

Each router has a corresponding fixed IP address on the interface that is participating in the VRRP Group.

For the VRRP feature to work correctly, the following conditions must be true:

- NNM must *not* attempt to discover and manage the virtual IP address of a VRRP Group.
- NNM *must* manage the fixed IP address of router interfaces in the VRRP Group.

NNM meets both of those conditions, internally tracking the virtual IP address of a VRRP group while managing the fixed IP address of router interfaces in the VRRP Group.

Checking NNM for Correct Handling of VRRP Virtual IP Addresses

To make sure NNM is configured for correctly handling VRRP virtual addresses, use the following procedure:

1. As `Administrator` or `root`, edit the following file:
   - *UNIX*:

     ```
     $OV_LRF/netmon.lrf
     ```
   - *Windows*:

     ```
     %OV_LRF%\netmon.lrf
     ```
2. Look for the following text within the file and verify that the bold text is present.

   `OVs_YES_START:ovtopmd,pmd,ovwdb:-P -k segRedux=true` **`-k migrateHSRPVirtualIP=true`** `:OVs_WELL_BEHAVED:15:PAUSE`
3. If the bold text is present, then NNM is handling the VRRP virtual IP address correctly.

4 If the bold text is not present add the text and save the file. You also need to restart the `netmon` process as follows. As `Administrator` or `root`, do the following:

   a Run the `ovstop netmon` command.

   b Run the `ovaddobj netmon.lrf` command.

   c Run the `ovstart netmon` command.

VRRP Discovery with Pre-existing NNM Topology

If you already have an NNM topology database (for example, if you have installed NNM and proceeded with an automatic discovery of your network), and the contents of the `netmon.lrf` shows `migrateHSRPVirtualIP=false` or does not contain the `migrateHSRPVirtualIP parameter,` then the database is likely to contain the virtual IP addresses of your VRRP groups. This is especially true if you used the auto-discovery feature.

You have two tasks:

1 Verify that NNM is correctly managing VRRP virtual IP addresses in your environment.

2 Remove from NNM any VRRP virtual IP addresses that NNM currently manages.

The next sections explain the procedure you should follow.

Verifying that NNM is Properly Managing VRRP Virtual IP Addresses

Follow the procedure shown in Checking NNM for Correct Handling of VRRP Virtual IP Addresses on page 279.

Remove Virtual IP Addresses from the NNM Topology

Next you need to remove any virtual IP addresses from the existing NNM topology data. There are a couple of common ways to do this:

- Use the following command to completely delete the router from the database (deleted routers get re-discovered in the next step, but without their virtual interfaces):

  — *UNIX*:

  ```
  $OV_BIN/ovtopofix -r <router>
  ```

  — *Windows*:

  ```
  %OV_BIN%\ovtopofix -r <router>
  ```

  You should remove the *node* (using the NNM ID) rather than the IP address. The NNM ID can be obtained with the following command:

  — *UNIX*:

  ```
  $OV_BIN/ovtopodump <router>
  ```

  — *Windows*:

  ```
  %OV_BIN%\ovtopodump <router>
  ```

  See the *ovtopofix(1M)* and *ovtopodump(1M)* reference pages for more information. Details about how to access reference pages are in the online help.

- Alternatively, from the NNM graphical interface (`ovw`), find each of your VRRP routers. Open each one to show all its interfaces. Delete the interface that corresponds to the virtual IP address.

After you have removed the virtual IP addresses from the NNM topology data, you need to rediscover the router.

Validating Your Results

To verify that your VRRP discovery is correct, restart Extended Topology discovery. See Start Extended Topology Discovery on page 23 for more information.

When the discovery finishes, open the VRRP Group Detail screen for each group and check for the following:

- If the VRRP virtual IP address shows up in both the IP Interface column and the Group Membership column, NNM is most likely managing the virtual IP address. You need to remove this interface from NNM (see Remove Virtual IP Addresses from the NNM Topology on page 274).
- You may see a VRRP Group with only one router, even though a VRRP Group must have at least two routers. This could have several causes. To resolve the problem, do the following:
  - Ensure that the actual IP address of the missing VRRP router is in NNM by running the following command:
    - *UNIX*:

      ```
      $OV_BIN/ovtopodump <fixed IP address>
      ```

    - *Windows*:

      ```
      %OV_BIN%\ovtopodump <fixed IP address>
      ```

  - Make sure that NNM has SNMP access to the missing router
  - Rerun discovery when all the VRRP router interfaces are up

    Note that if a VRRP router interface is down during discovery, the Advanced Routing SPI will discover VRRP groups incompletely. This is because the VRRP MIBs at the time of discovery reflect the actual state. This should be a transient problem that resolves when an Extended Topology discovery occurs after the interface is back up.

Understanding VRRP View Status Information

The Active Problem Analyzer polls VRRP routers and reports VRRP group status information. See Table 22 for a list of the various group status colors and their meanings.

Table 22 VRRP Group Status

VRRP Group Status Color	VRRP Group Status Description
Unknown (blue)	The VRRP group has not been polled since discovery completed, so the state of the group has not been determined yet. The actual state of this VRRP group will be displayed as soon as it is polled.
Normal (green)	The VRRP group is correctly configured and operational. This VRRP group is providing routing functionality, and all standby routers are available.
Warning (cyan)	The VRRP group has an interface in the `Active` state, and an interface in the `Standby` state, but it also has at least one interface that is not in the `Listen` state. This VRRP group is providing routing functionality. However, one or more standby routers are definitely unavailable.
Minor or Marginal (yellow)	The VRRP Group has an interface in the `Active` state, but there is no interface in the VRRP group which is in the `Standby` state. This VRRP group is providing routing functionality, but *there is no standby router available*.
Major (orange)	Multiple interfaces in the VRRP group are in the `Active` state, or multiple interfaces in the VRRP group are in the `Standby` state. This VRRP group is *not functioning correctly*. There is almost certainly a problem with routing functionality.

Table 22 VRRP Group Status

VRRP Group Status Color	VRRP Group Status Description
Critical (red)	The VRRP group has no interface in the `Active` state. This VRRP group is *definitely not* providing routing functionality.

The Active Problem Analyzer displays the following alarms in the alarms browser.

- `OV_HSRP_No_Active`: The reported VRRP group is completely inoperational.
- `OV_HSRP_Multiple_Active`: The reported VRRP group is in an abnormal condition as there are multiple active interfaces.
- `OV_HSRP_No_Standby`: The reported VRRP group has no standby interface.
- `OV_HSRP_Degraded`: The reported VRRP group has an interface that is not functioning properly
- `OV_HSRP_FailOver`: The reported VRRP group has changed its active interface.
- `OV_HSRP_Standby_Changed`: The reported VRRP group has changed its standby interface.
- `OV_HSRP_Normal`: The reported VRRP group is now functioning normally.

See the *trapd.conf* reference page for more information. Details about how to access reference pages are in the online help.

Potential Status Anomalies

Subtle conditions can lead the Advanced Routing SPI to yield unexpected status for VRRP groups.

- If you run an Extended Topology discovery while any VRRP group interfaces are down, Extended Topology will not discover all of the VRRP group interfaces and will calculate and display VRRP group status using incomplete information. After you run an Extended Topology discovery that finds all of the VRRP group's router interfaces, Extended Topology will show accurate VRRP status.

⚠ As you add equipment with new VRRP virtual IP addresses to your managed environment, you need to add these addresses to the `netmon.noDiscover` file prior to turning on the new equipment. See the reference page for *netmon.noDiscover* for usage information. Details about how to access reference pages are in the online help. If you fail to do this, and NNM discovers these virtual IP addresses, you will need to complete the tasks explained in VRRP Discovery with Pre-existing NNM Topology on page 280.

Using the VRRP View to Diagnose Network Problems

The following scenario gives one example of using Extended Topology information to diagnose a network problem with VRRP. This discussion assumes you are familiar with VRRP.

Suppose you have two directly connected router interfaces, interface a on router 1 and interface b on router 2. This discussion refers to these router interfaces as interfaces 1.a and 2.b respectively.

Assume both interfaces are located within the same LAN segment. You have router interface 1.a configured as `active` and router interface 2.b configured as `standby` for standby group a.b.c.d. a.b.c.d represents the virtual IP address of the standby group.

In this example, interface 1.a fails, and the following situation occurs:

1 Interface 2.b becomes active and generates an `OV_HSRP_FailOver` alarm.

2 To investigate this alarm, you open a VRRP Group view by selecting the `OV_HSRP_Failover` alarm in the Alarm Browser and use the `Actions:Views-> VRRP Group Detail` menu.

3 The `VRRP Group Detail` view shows you that interface 2.b is now active and interface 1.a is down. You need to troubleshoot interface 1.a.

Running OSPF Discovery

Running OSPF discovery results in the Advanced Routing SPI discovering OSPF information about your network.

During OSPF discovery, the Advanced Routing SPI reads information from one of two OSPF version 2 MIBs: RFC1850 or RFC1253. Routing devices must support one of these MIBs for the Advanced Routing SPI to read OSPF information from them.

During the OSPF discovery, the Advanced Routing SPI discovers which area OSPF devices are located in, and how these areas relate to one another.

You must manually run OSPF discovery as outlined in the following procedure:

1 Edit the following file using the guidelines listed below:

   - *UNIX*:

     ```
     $OV_CONF/nnmet/Ospf.cfg
     ```

   - *Windows*:

     ```
     %OV_CONF%\nnmet\Ospf.cfg
     ```

   There are OSPF configuration instructions included within the `Ospf.cfg` file. Use the following guidelines to configure the `Ospf.cfg` file.

   - Add the IP address of a router being managed by NNM to seed OSPF discovery.
   - You can set the OSPF discovery range using either an OSPF area INCLUDE or EXCLUDE list, but not both, as shown in the `Ospf.cfg` file.
   - If you use an INCLUDE list, the Advanced Routing SPI discovers only the areas in the INCLUDE list.
   - If you use an EXCLUDE list, the Advanced Routing SPI discovers all areas except those on the EXCLUDE list.

2 Run the following script:

- *UNIX*:

  ```
  $OV_BIN/ospfdis.ovpl
  ```

- *Windows*:

  ```
  %OV_BIN%\ospfdis.ovpl
  ```

You must run `ospfdis.ovpl` each time you modify the `Ospf.cfg` file for OSPF discovery changes to affect the OSPF view.

3 After OSPF discovery completes with no errors shown on your screen, check for errors in the following file:

- *UNIX*:

  ```
  $OV_PRIV_LOG/ospfdis.err
  ```

- *Windows*:

  ```
  %OV_PRIV_LOG%\ospfdis.err
  ```

4 If there are errors in the `ospfdis.err` file, fix any errors that may impact any devices or areas you are interested in viewing and rerun the `ospfdis.ovpl` command.

5 Repeat step 4 until you are satisfied with your OSPF view.

Troubleshooting OSPF Discovery

Someone knowledgeable about OSPF and the discovered environment should remedy problems found in the `ospfdis.err` file. Some common OSPF discovery errors are:

- Device access problems: Device SNMP community strings are missing or incorrect. Find and add the device community names using the `SNMP Configuration` menu item from the NNM `Options` menu. This could be the source of numerous `ospfdis.err` file entries.
- Seed device is inaccessible: Make sure the seed device you add to the `Ospf.cfg` file has been discovered by NNM and is accessible from the system that the Advanced Routing SPI is running on.
- Discovered device is inaccessible: Make sure the device has been discovered by NNM and is accessible from the system the Advanced Routing SPI is running on.

- No seed in the `Ospf.cfg` file: Edit the `Ospf.cfg` file and add a seed router IP address.
- The Advanced Routing SPI expecting IP addresses: Make sure the seed router IP address is valid.
- Using both INCLUDE and EXCLUDE area lists: Make sure you have not configured both an INCLUDE and an EXCLUDE area list.
- INCLUDE or EXCLUDE area list errors: Make sure you configure the lists correctly in the `Ospf.cfg` file.

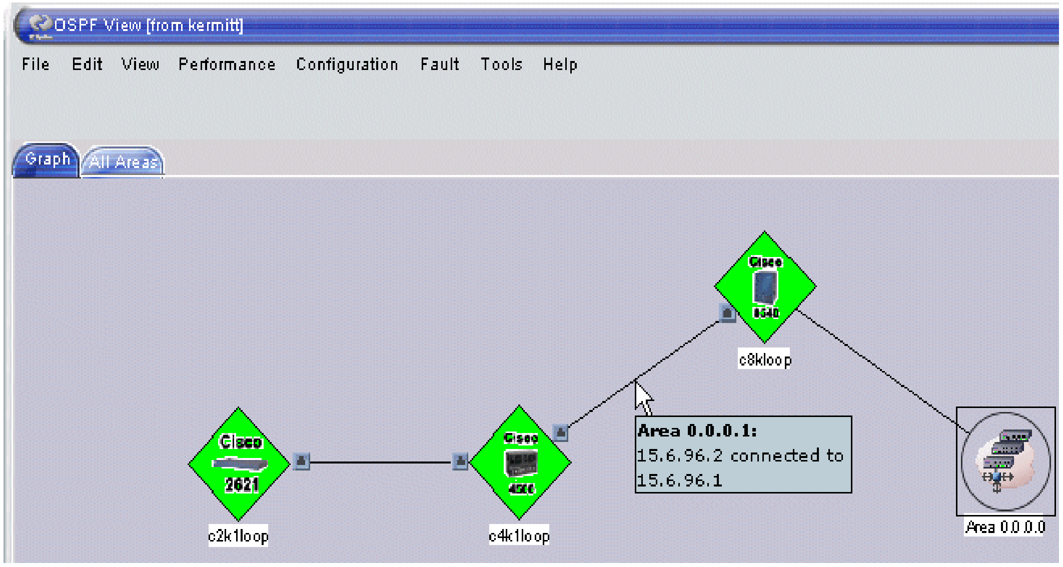
Using the OSPF View to Review Network Configurations

The following scenario shows an example of using the Advanced Routing SPI information to investigate an OSPF configuration question.

Suppose you wanted to make sure the Area Border Routers you set up for your network were configured correctly. You know you configured interfaces on c8kloop to act as the Area Border Routers for both `Area 0.0.0.0` and `Area 0.0.0.1`.

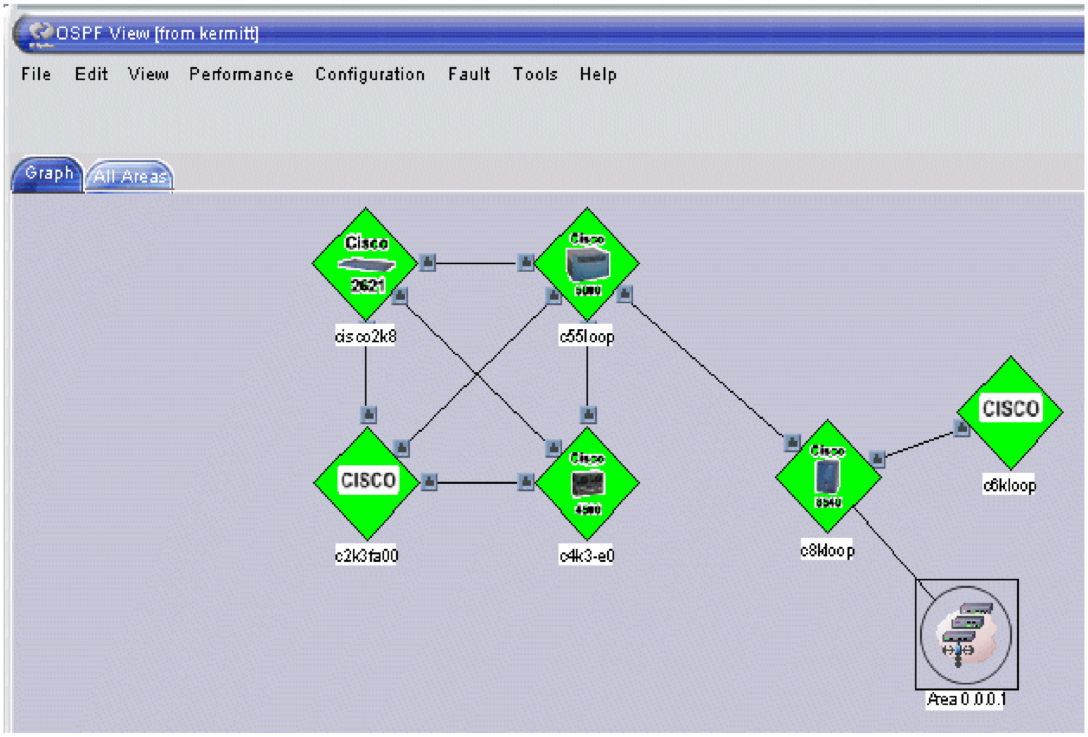
To check this, you could open an OSPF view from Home Base and investigate. Figure 22 and Figure 23 show both areas configured as you designed them.

Figure 22 Checking the Area Border Router of Area 0.0.0.1



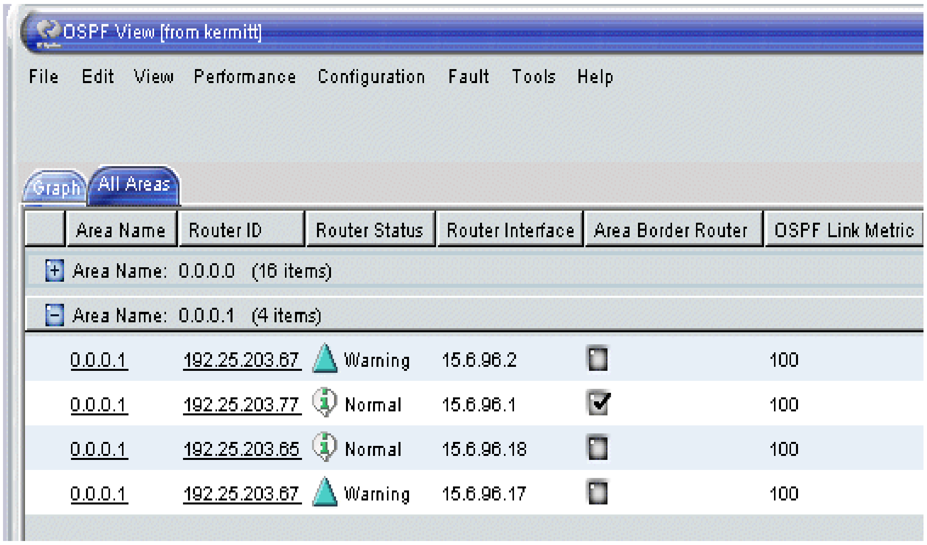
Area 0.0.0.1:
15.6.96.2 connected to
15.6.96.1

Figure 23 Checking the Area Border Router of Area 0.0.0.0



To review additional information about each area, you can select the `All Areas` tab. Figure 24, shows some of the additional information that is available. You can use this view to research additional information such as the OSPF link metric and router status.

Figure 24 Additional Information from the `All Areas` Tab



	Area Name	Router ID	Router Status	Router Interface	Area Border Router	OSPF Link Metric
+	Area Name: 0.0.0.0 (16 items)					
−	Area Name: 0.0.0.1 (4 items)					
	0.0.0.1	192.25.203.67	Warning	15.6.96.2	☐	100
	0.0.0.1	192.25.203.77	Normal	15.6.96.1	✓	100
	0.0.0.1	192.25.203.65	Normal	15.6.96.18	☐	100
	0.0.0.1	192.25.203.67	Warning	15.6.96.17	☐	100

Running IPv6 Discovery

Before starting your IPv6 discovery, you need add some information to three files as described below.

The following five files control your IPv6 discovery and status polling:

- *UNIX*:

  ```
  $OV_CONF/nnmet/IPv6.conf
  $OV_CONF/nnmet/IPv6Polling.conf
  $OV_CONF/nnmet/IPv6Prefix.conf
  $OV_CONF/nnmet/IPv6Scope.conf
  $OV_CONF/nnmet/IPv6Seed.conf
  ```

- *Windows*:

  ```
  %OV_CONF%\nnmet\IPv6.conf
  %OV_CONF%\nnmet\IPv6Polling.conf
  %OV_CONF%\nnmet\IPv6Prefix.conf
  %OV_CONF%\nnmet\IPv6Scope.conf
  %OV_CONF%\nnmet\IPv6Seed.conf
  ```

You configure these files to adjust discovery and polling to meet your needs. Each file contains configuration examples and instructions showing how to add information. Use the following procedure to complete your IPv6 discovery:

1. Modify the `IPv6Seed.conf` file.

   The Advanced Routing SPI uses the `IPv6Seed.conf` file to seed your IPv6 discovery. For best results, enter the addresses of all of the routers and end nodes you want to discover into this file. To enter these addresses, follow the instructions included within the `IPv6Seed.conf` file.

2. This step is not mandatory. The Advanced Routing SPI reads routing tables from routers and can discover devices beyond the nodes specified in the `IPv6Seed.conf` file. If you want to limit IPv6 discovery to the nodes you add to this file, edit the `IPv6.conf` file and follow the instructions contained in the file.

3. Modify the `IPv6Prefix.conf` file.

   IPv6 prefix groups are similar to subnets in IPv4. The `IPv6Prefix.conf` file is used to give a user-friendly name to your prefix groups after they are discovered.

4 This step is not mandatory. The Advanced Routing SPI allows you to list prefix groups and IPv6 addresses that you either want discovered or want excluded from discovery. To enter these addresses or prefix groups, follow the instructions included within the `IPv6Scope.conf` file.

5 This step is not mandatory. You can modify the `IPv6Polling.conf` file if you need to change the polling frequency.

  The `IPv6Polling.conf` file is used to modify the polling frequency of nodes.

6 If you have not enabled Extended Topology yet, you must enable Extended Topology to start your first discovery. See Extended Topology Discovery on page 22 for more information.

7 If Extended Topology was enabled before you configured your IPv6 discovery, restart Extended Topology discovery. See Start Extended Topology Discovery on page 23 for more information.

➤ IPv6 discovery relies heavily on both forward and reverse name resolution. You can achieve name resolution by using either a DNS server or hosts files. Make sure all of your IPv6 addresses resolve properly before proceeding.

Execute the `IPv6NameByAddr` command as follows to check reverse name resolution:

- *UNIX*:

```
/opt/OV/support/NM/getIPv6NameByAddr <IPv6Addr>
```

- *Windows*:

  The support directory and tools are located on the product CD.

See Troubleshooting IPv6 Discovery on page 293 for more information about diagnosing IPv6 discovery problems.

Troubleshooting IPv6 Discovery

Some common IPv6 discovery errors are listed below:

- Unexpected nodes show up in your map: The Advanced Routing SPI discovers IPv6 nodes beyond the seed file entries. Extra IPv6 nodes will show up if they are found in the router’s tables.

- Some nodes are missing from the map: Make sure these nodes are entered in the following file:
  - *UNIX*:

    `$OV_CONF/nnmet/IPv6Seed.conf`
  - *Windows*:

    `%OV_CONF%\nnmet\IPv6Seed.conf`

  This is the only way to ensure that the Advanced Routing SPI will discover these nodes. Make sure you enter only valid nodes in the `IPv6Seed.conf` file.
- Extended Topology shows a node’s status as being down when you know the node is up: Extended Topology relies on the Advanced Routing SPI’s IPv6 ping for all its status. Make sure you can IPv6 ping all of the node’s IPv6 addresses from the management station.
- The map shows your end nodes labeled with IPv6 Addresses rather than names: Make sure that both forward and reverse DNS are working for the node from the management station.
- Your routers are showing up as end nodes instead of routers: Make sure this is a supported router with IPv6 MIBs loaded. Also make sure you have SNMP access properly configured.
- Your map shows prefix groups connected to the router but no end nodes in the prefix groups. There are several possible solutions listed below:
  - You may have the `End Nodes` check box deselected in the view.
  - The router may be the only node in the prefix group. This is common for routers. Many times routers have prefix groups configured but no end nodes in them yet.
  - There may be very little traffic on the network. To remedy this, specify nodes in the following file for that prefix group:
  - *UNIX*:

    `$OV_CONF/nnmet/IPv6Seed.conf`
  - *Windows*:

    `%OV_CONF%\nnmet\IPv6Seed.conf`

- Your topology contains many disconnected nodes: Make sure you can access your router via SNMP from the management station. Also, make sure the IPv6 routers you are discovering support the IPv6 MIBs.
- Your topology looks incorrect on the map. There are several possible solutions listed below:
  - Reload the page. If you bring up the page while a lot of status events are being generated, the topology shows up incorrect.
  - Make sure the prefix length is correct for the addresses in the seed file.

Displaying Routers Not Supporting IPv6 MIBs

If you attempt to discover a router that does not support the IPv6 MIBs, the router can show up as two different nodes. To try and show the router properly, take the following steps:

1. Ensure that all IPv6 interface addresses are registered on your DNS server with the same name.
2. Ensure that at least one IPv4 interface of the router is also registered on your DNS server under the same name.
3. Ensure that you have configured the SNMP community strings in NNM for the IPv4 interface of the router.
4. Ensure that all IPv6 addresses and prefix lengths for the router are listed in the following file without specifying an optional name.
   - *UNIX*:

     ```
     $OV_CONF/nnmet/IPv6Seed.conf
     ```

   - *Windows*:

     ```
     %OV_CONF%\nnmet\IPv6Seed.conf
     ```

5. Identify all end nodes located beyond the router that you wish to manage and enter their addresses in the `IPv6Seed.conf` file since these end nodes will not be discovered automatically. Do not enter the link-local addresses of these nodes.

Displaying IPv6 Nodes with Multiple Addresses

If you configure a node with more than one IPv6 address you should list all of the IPv6 addresses for this end node in the following file:

- *UNIX*:

  ```
  $OV_CONF/nnmet/IPv6Seed.conf
  ```

- *Windows*:

  ```
  %OV_CONF%\nnmet\IPv6Seed.conf
  ```

Make sure that your IPv6 addresses are configured as follows:

1. Make sure that all IPv6 addresses for this node are registered on your DNS server with the same name.
2. Make sure that all IPv6 addresses for this node are listed in the seed file without specifying an optional name.

Viewing IPv6 Information

You can access an index of all the NNM and Extended Topology views by pointing your web browser to the following URL:

```
http://hostname:7510
```

From this index you can access the `IPv6 Network View` and the IPv6 information.

There are several other ways to view IPv6 Information. See Access Dynamic Views on page 104 for additional information.

IPv6 Network View

If you select the `IPv6 Network View` from Home Base, the Advanced Routing SPI displays a global map showing all discovered IPv6 global prefix groups and the nodes logically connected to each prefix group.

The `IPv6 Network View` displays nodes having *Global addresses*, also called aggregatable global unicast addresses. These addresses refer to IPv6 addresses that begin with 001.

The Advanced Routing SPI groups IPv6 nodes according to prefix groups, which are similar to subnets in IPv4. If your DNS is resolving names properly, you will see nodes with names rather than addresses. If DNS is not resolving names properly, your views will contain one of the node's IPv6 addresses.

You can move your mouse over a node to display additional information. If you double-click on a node, the Advanced Routing SPI displays details about that node. Node status is updated dynamically. See Understanding IPv6 Status Information on page 300 for more information on how the Advanced Routing SPI determines node status.

The default setting for `Scope` has both the `Global` option button and the `Include End Nodes` check box selected. If you want to restrict the view to network nodes without end nodes, uncheck the check box and select `[Refresh]`.

If you want to see only site-local nodes, click the `Site-local` radio button and select [Refresh]. For more information see IPv6 Site-Local Network View on page 299.

If you use IPv4 compatible addresses, be aware that IPv4 compatible status is not shown in the table. You can select any of the interface links to get all the related status information for any of the interfaces.

Prefix Groups

Along with routers and end-nodes, the Advanced Routing SPI allows Extended Topology to display prefix groups. Prefix groups are similar to subnets in IPv4. All nodes that have addresses belonging to a specific prefix group are shown as being connected to that prefix group's icon.

The Advanced Routing SPI allows you to name your prefix groups in the following file in order to provide better segmentation of your IPv6 views.:

- *UNIX*:

  ```
  $OV_CONF/nnmet/IPv6Prefix.conf
  ```

- *Windows*:

  ```
  %OV_CONF%\nnmet\IPv6Prefix.conf
  ```

See Running IPv6 Discovery on page 292 for more information on configuring your prefix group names.

➤ Link-local addresses are not assigned a prefix group and will not show up in the map unless their parent nodes also have a global or site-local address. The Advanced Routing SPI never polls link-local addresses for status.

IPv6 nodes can have more than one address per interface. In IPv6 views, connection lines between nodes and prefix groups do not always represent a one-to-one mapping to interfaces on a node. A single interface can have multiple addresses which may be in two different prefix groups.

Refer to Table 23 when reviewing the following possible examples:

- If a node has one interface and that interface has one global address, the Advanced Routing SPI displays only one line connecting the node to one prefix group.
- If a node has one interface and has two global addresses within the same prefix group, the Advanced Routing SPI displays only one line connecting the node to one prefix group.
- If a node has one interface and has two global addresses with two different prefixes, the Advanced Routing SPI displays two lines connecting the node to two different prefix groups.

- If a node has two interfaces and a total of four global addresses belonging to three different prefix groups, the Advanced Routing SPI displays three lines connecting the node to three different prefix groups.

Table 23 Prefix Group Scenarios

Description	Node Count	Interface Count	Global Address Count	Prefix Count	Expected Result
A node has one interface and that interface has one global address.	1	1	1	1	The node has one line going to the prefix group symbol.
A node has one interface and the interface has two global addresses within the same prefix.	1	1	2	1	The node has one line going to the corresponding prefix group symbol.
A node has one interface and the interface has two global addresses with different prefixes.	1	1	2	2	The node has two lines going to the two prefix group symbols.
A node has two interfaces and a total of four global addresses belonging to three different prefixes.	1	2	4	3	The node has three lines going to three prefix group symbols.

The status of a prefix group is compounded from the status of the addresses belonging to the prefix group. For example, suppose a prefix group icon is colored blue. This means that there is an address that is not responding to IPv6 pings within that prefix group. See Understanding IPv6 Status Information on page 300 for more information on understanding IPv6 node status.

IPv6 Site-Local Network View

From Home Base, if you select the `IPv6 Network View`, then select a `Site Local Network View`, the Advanced Routing SPI displays a site-local map, showing all discovered IPv6 site-local prefix groups and the nodes logically connected to each prefix group.

If you double-click on a site-local prefix group, you can view the nodes contained within that site-local prefix group. You can double-click on a prefix group and view information about each of the nodes in that prefix group.

Understanding IPv6 Status Information

To monitor the address status of a device, the Advanced Routing SPI uses an IPv6 ping rather than using SNMP requests. The Advanced Routing SPI considers a device to be down if it does not respond to an IPv6 ping.

You can modify the following file to adjust the IPv6 ping frequency:

- *UNIX*:

  ```
  $OV_CONF/nnmet/IPv6Polling.conf
  ```

- *Windows*:

  ```
  %OV_CONF%\nnmet\IPv6Polling.conf
  ```

➤ The Advanced Routing SPI does not ping link-local addresses and marks any link-local address with an UNKNOWN status.

IPv6 site-local and global addresses have three status possibilities: normal, critical, or unknown. The Advanced Routing SPI derives interface status from address status and node status from interface status. From these address status, we derive interface status. The Advanced Routing SPI also derives prefix group status from address status. See Table 24 for a summary of how the Advanced Routing SPI compounds IPv6 status information.

Table 24 Deriving IPv6 Status

Address Status	Interface Status	Node Status
Site-local address status	Site-local interface status	Site-local node status
Global address status	Global interface status	Global node status
Site-local plus global address status	Overall interface status	Overall node status

The status of an IPv4-compatible address contributes to the overall interface status it is assigned to. This can also impact the overall node status.

The Advanced Routing SPI derives interface status using the compounding rules show in Table 25.

Table 25 Compound IPv6 Address Status into Interface Status

Current Address Status	Compounded Interface Status
Any address status is down.	Down
All address statuses are unknown.	Unknown
At least one address status is up and all other address statuses are unknown.	Normal

The Advanced Routing SPI derives interface status using the compounding rules show in Table 26.

Table 26 Compounding IPv6 Address Status

Condition of Contributing Objects	Compounded Status
No Objects Exist	Unknown (blue)
All are unknown	Unknown (blue)
All are up (except those that are unknown)	Normal (green)
One is down and all others are up (except those that are unknown)	Warning (cyan)
More than one is up and more than one is down (except the ones that are unknown)	Minor/Marginal (yellow)
One is up and all others are down (except the ones that are unknown)	Major (orange)
All are down (except the ones that are unknown)	Critical (red)

See the *Understanding IPv6 Status* section of the online help for more information.

Understanding IPv6 Events

The Advanced Routing SPI generates many new IPv6 events. Most of these events are logged and not forwarded to the NNM Alarm Browser. You can open an IPv6 view from events located in the NNM Alarm Browser, by selecting `Actions`->`Views`->IPv6.

The Advanced Routing SPI displays one event, `OV_IPV6_Address_DOWN,` in the NNM Alarm Browser. NNM includes both the OV_IPV6_Address_DOWN and `OV_IPV6_AddressUP` events in the Pairwise event correlation. `F`or more information about the behavior of the NNM Event Correlation System and the Pairwise event correlation, see *Managing Your Network with HP OpenView Network Node Manager*. For more information about events and their definitions, see the *trapd.conf* manpage.

9 Overlapping Address Domains

Discover and Monitor Nodes with Overlapping Address Domains

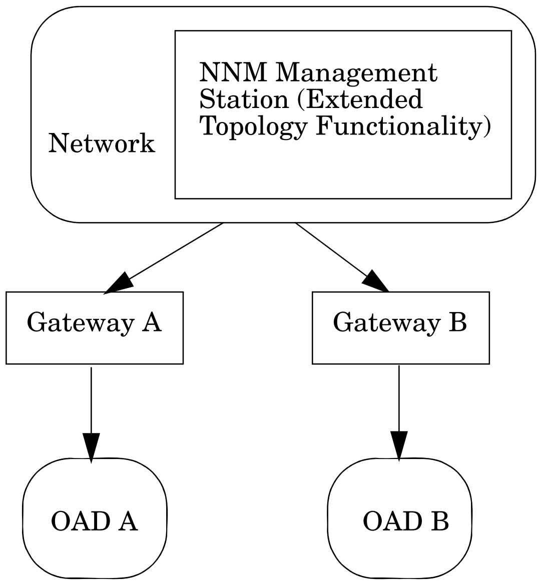
When you use NNM to manage multiple domains that use private IP addresses, NNM often discovers duplicate private IP addresses. To resolve this situation, you need to use the Extended Topology's Overlapping Address Domain functionality.

You can use Extended Topology to distinguish your overlapping addresses by configuring each address group into an Overlapping Address Domain (OAD). A single OAD is a set of IPv4 addresses that are static, do not overlap, and are directly routable without manipulation of the IPv4 header.

Figure 25 on page 304 shows an example of Extended Topology managing two private address domains from a point outside of both domains If the IP addresses in OAD A and OAD B overlap, you can configure Extended Topology to differentiate addresses from the two overlapping address domains.

If there is a firewall between the NNM management station and any nodes tied to a specific OAD, you must configure this firewall to pass ICMP (port 7), SNMP (port 161), and SNMPTRAP (port 162) packets between the management station and the managed nodes. If you want to log on to any OAD nodes (telnet) using Dynamic Views from the management station, you will need to open port 23 as well. This minimal relaxation of the firewall is required to support network management through the firewall. By restricting communications to the management station only, there is virtually no loss of security.

Figure 25 Overlapping Address Domains



Configure Extended Topology to Discover and Monitor Overlapping Private Internet Addresses

Before you can use Extended Topology to monitor your overlapping private IP addresses, you need to provide the information discussed in this section.

➤ The OAD functionality is available only when Extended Topology is enabled. As a result, OAD nodes must be placed in the netmon.noDiscover file and removed from the NNM classic database. See the reference page for *netmon.noDiscover* for usage information. Details about how to access reference pages are in the online help. For information on removing nodes from the NNM classic database, see the *Managing Your Networks* manual.

Provide Extended Topology with OAD Information

A key concept of monitoring overlapping private IP addresses is understanding the OAD. The OAD denotes a set of unique, private IP addresses. For example, an OAD might represent the private IP addresses of a small business, specific department, or a specific workgroup in a large company.

1 For each OAD that you define, you need to create a separate directory under the following directory:

- *UNIX*:

```
$OV_CONF/nnmet/dupip
```

- *Windows*:

```
%OV_CONF%\nnmet\dupip
```

There is no specific naming convention so you can choose a friendly name for each directory. For example, if you have a group of private IP addresses in an OAD for a store called *My Shop*, you could name the directory *myShopDomain*.

2 Within each new directory, you need to create a `dupip.conf` file and add commands to this file to define the OAD. For example, if you named your directory myShopDomain in step 1, you would create the following file for this OAD:

- *UNIX*:

  ```
  $OV_CONF/nnmet/dupip/myShopDomain/dupip.conf
  ```

- *Windows*:

  ```
  %OV_CONF%\nnmet\dupip\myShopDomain\dupip.conf
  ```

  ➤ In the `dupip` directory, there is a sample `dupip.conf` file. Refer to this file for examples and instructions on how to create your new `dupip.conf` file.

3 In the same directory as your new `dupip.conf` file, create a file named `dupip.seed`. This file lists the management IP addresses (required) and node names (optional) that you wish to manage in this OAD. Add the IP addresses in the first column and the optional node names in the second column. Enter one IP address and node name per line as described in the `dupip.conf` file.

If you choose to provide node names, your name resolution scheme must be able to resolve each node name in `dupip.seed` to the IP address given for it.

⚠ Do not add the virtual IP address of an HSRP group to the `dupip.seed` file.

The new seed file defines the discovery zone for the OAD, which appears in the `Overlapping Address Domains` tab of the Extended Topology configuration interface.

4 After configuring your `dupip.conf` and `dupip.seed` files, run the `ovdupip` command to make sure your file entries are syntactically correct. If there are errors in the files, this tool will tell you what is wrong and where to look to remedy the problem. See the *ovdupip* reference page for more information. Details about how to access reference pages are in the online help.

5 Open the Extended Topology Configuration web page. See Open the Extended Topology Configuration Page on page 43 for more information.

6 Select the **Overlapping Address Domains** tab.

7 Select **Refresh Configuration and Activate Changes** to read, test, and deploy any changes you made. If Extended Topology determines your changes are free of errors, it creates a zone for every defined OAD. These changes or additions will affect the next discovery cycle.

8 To immediately initiate a complete discovery of all zones, restart Extended Topology discovery. See Start Extended Topology Discovery on page 23 for more information.

9 If you add new devices to or delete devices from a single OAD (zone), you can save time by initiating an Extended Topology discovery on that single zone. See Discover a Single Zone on page 35 for more information.

Delete a Single OAD Zone

You can remove information about a single OAD zone without having to rediscover all of the other zones. To do this, use the following procedure:

1 Go to the Extended Topology Configuration web page. See Open the Extended Topology Configuration Page on page 43 for more information.

2 Select the **Overlapping Address Domains** tab.

3 Select the option button to the left of the OAD zone you want to delete.

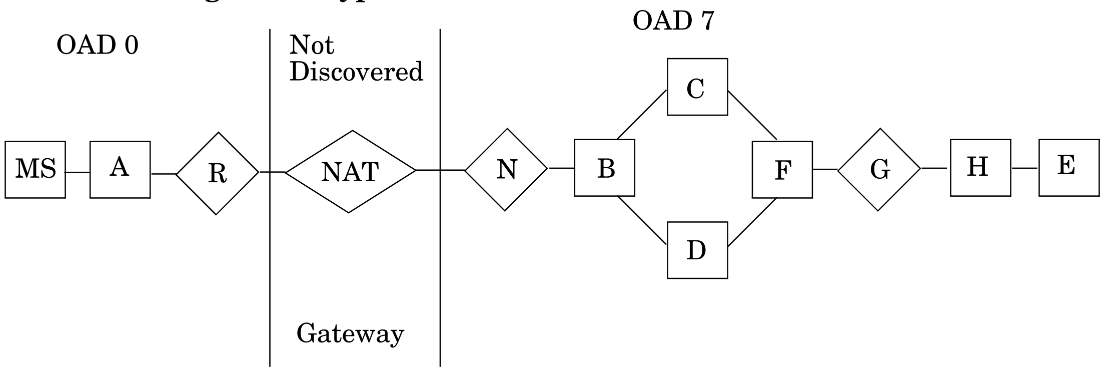
4 Select **Delete OAD**.

Understand OAD Discovery

Figure 26 on page 308 shows a typical OAD network where the management station (MS) connects to a small number of devices (devices A and R) and to a NAT (Network Address Translation) device.

Devices on the left side of the NAT device are in OAD 0. Devices on the right side of the NAT are in a non-zero OAD Domain (OAD 7 in Figure 26) and may correspond to a specific ISP customer. The OAD configuration uses the term Gateway to identify the NAT device. In addition, there can be more than one Gateway per OAD.

Figure 26 Typical OAD Environment



The NAT device translates a subset of private addresses to corresponding public translated addresses. This allows the MS to communicate to devices in OAD 0 using the non-translated private addresses and to communicate to devices in the other OAD (OAD 7) using the translated private addresses.

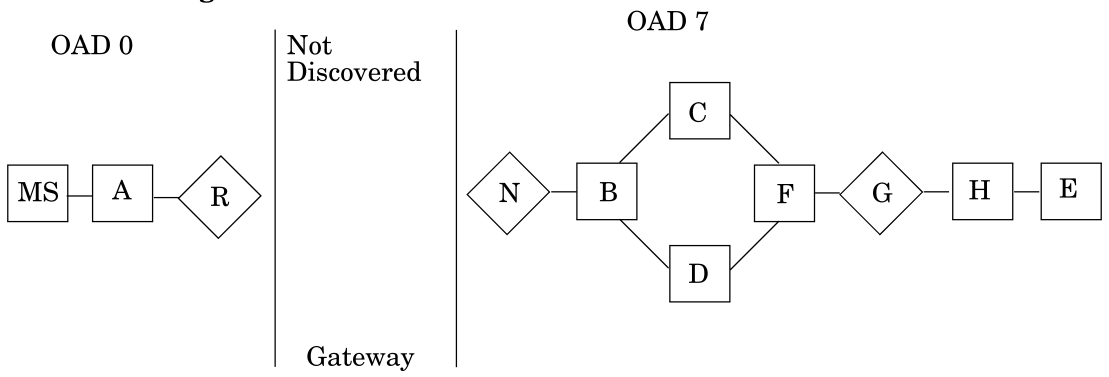
You can add entries to the `dupip.seed` configuration file, specifying one address per device to be used as the translated public management address for the device. Although other addresses may be translated by the NAT, Extended Topology will use this public management address for SNMP management and discovery. For more information on `dupip.seed` files, see step 3 on page 306.

For a specific device, when you decide which address to have Extended Topology translate and use as the management address, it is good practice to select the most upstream address or use the device's loopback address. Whenever possible, select an address that is available on an interface closest to the management station. This will facilitate better analysis because partial failures on the device will still allow the management station to talk to the device.

OAD and Undiscovered NAT Devices

Although Extended Topology uses the functionality of the NAT device for discovery and monitoring, it may not actually discover the NAT device. For example, organizations frequently deploy NAT devices for security or to isolate different parts of the network. If these NAT devices do not respond to MIB-2 SNMP queries, Extended Topology does not discover them. As a result, the NAT device may not be in the Extended Topology or may not be connected properly. When a NAT device is not discovered, the connectivity from the Extended Topology may look more like the diagram in Figure 27.

Figure 27 Undiscovered NAT Device

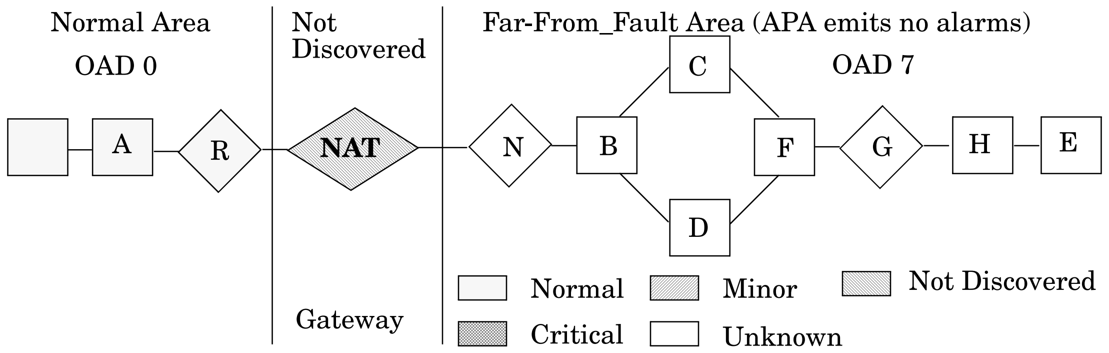


Potential APA Concerns About Undiscovered NAT Devices

If a failure occurs in the OAD environment, as shown in Figure 27, APA sets status and emits alarms consistent with user expectations unless the entire OAD environment becomes inaccessible.

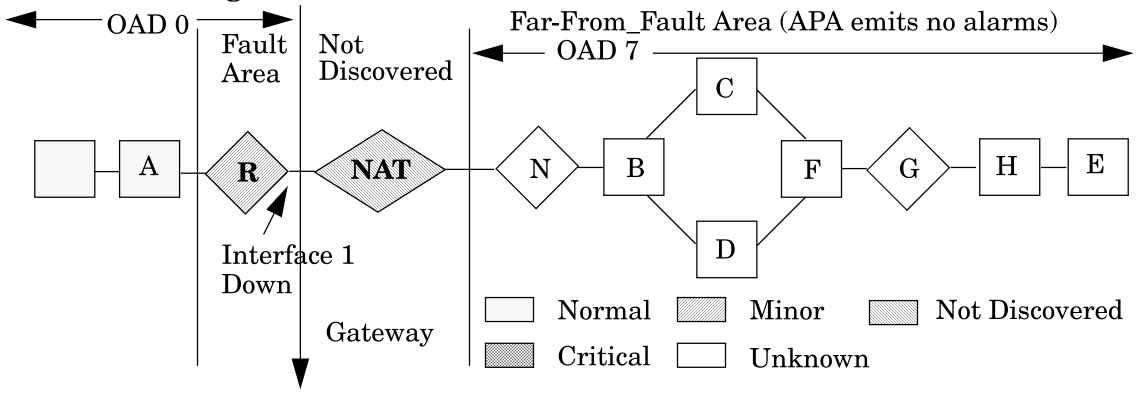
For example, if device N fails, APA considers all of the devices in the OAD 7 area to be in the Far-From-Fault Area as shown in Figure 28. In this example, APA does not generate any alarms and sets the status of devices in OAD 7 to unknown. This problem occurs because the fault area must contain at least one device that is accessible via SNMP or ICMP. Since device N is not connected to any accessible upstream device, APA places device N in the Far-From-Fault Area instead of the Fault Area.

Figure 28 OAD Scenario: Device N Fails



A similar result occurs if the NAT device fails or the downstream interface of device R fails, as shown in Figure 29. In either case, the entire OAD 7 area will be inaccessible. APA will not emit any alarms for OAD 7 because the entire area is Far-From-Fault. However, the downstream interface of device R will be in the Fault area. Therefore, APA sets the node status of device R to minor, sets the interface status to critical, and emits an `OV_APA_IF_DOWN R` alarm. See APA Failure Diagnosis on page 133 for more information.

Figure 29 OAD Scenario: NAT Device or Device R Interface Fails



Resolve APA Concerns About Undiscovered NAT Devices

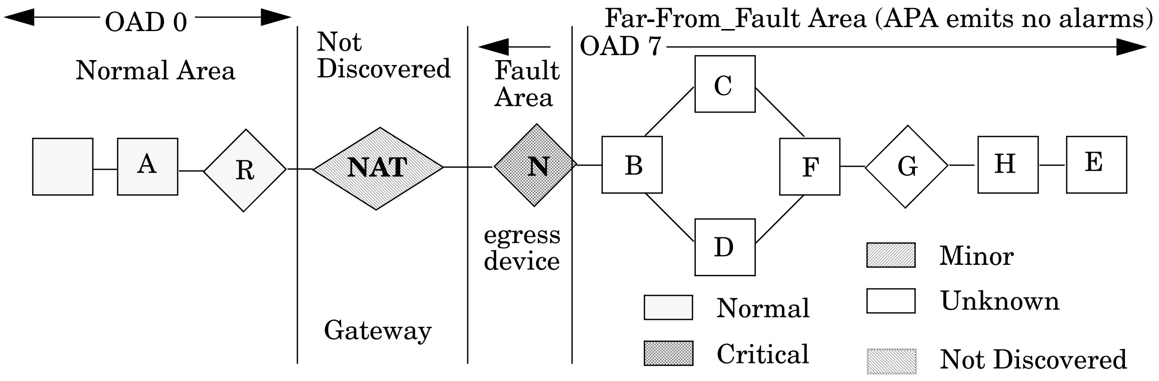
You can help APA accurately determine the root cause in OAD environments with undiscovered NAT devices. To do this, you need to configure Extended Topology to recognize a device located downstream from a NAT device.

The `dupip.conf` configuration file provides a mechanism, the **NextHop** keyword, to specify the device one hop downstream from a NAT device. In the example shown in Figure 30, the downstream device is device N. If device N has an IP Address of 10.1.2.3 and a public management address (NAT address) of 15.1.2.3, then the following entry in the `dupip.conf` file would specify this device: `NextHop IP=15.1.2.3`. This downstream device is sometimes referred to as the egress device (router or switch).

When you enter an IP address using the NextHop keyword, you must enter the device's public address. You cannot enter an address range or wildcard. If the NAT device maps several addresses associated with the egress router and Extended Topology discovers these addresses, the result is that several public addresses are associated with the egress router. However, there is still only one associated management address. In this situation, you must use the management address as the NextHop address that you add to the `dupip.conf` file.

You can find more information about using the NextHop keyword in the `dupip.conf` file.

Figure 30 OAD Scenario: Egress Device N Fails



When you specify one or more egress devices in the `dupip.conf` file using the NextHop keyword, APA processes the egress routers differently when they fail. For example, if an egress device fails, as shown in Figure 30, APA will do no further analysis on the specified egress device. Instead, APA immediately reports the egress device as a primary failure in the Fault Area, reports the egress device status as critical, and generates the appropriate status alarms.

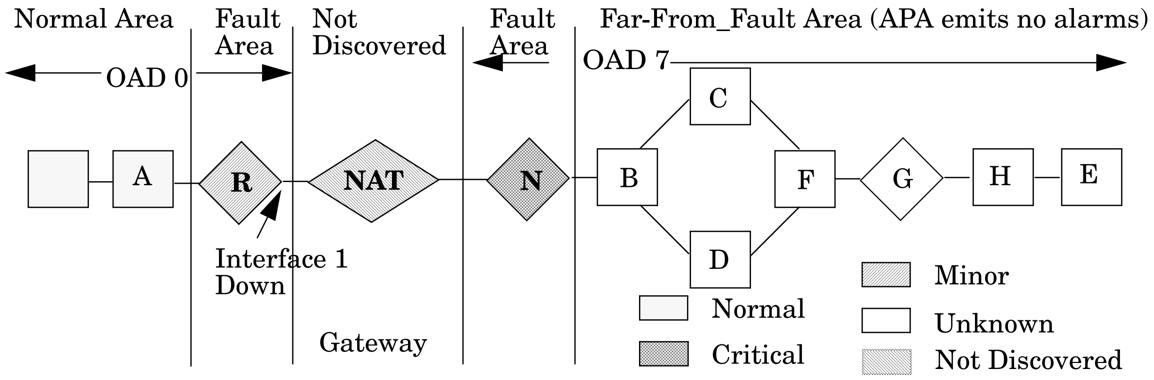
The result is that for a catastrophic failure where an entire OAD area is inaccessible, only one alarm is generated to indicate the device that failed (or a device nearby). Thus, the network operator and administrator will not see an Alarm Browser that is cluttered with alarms from downstream devices that are not actually critical. The operator only sees an alarm and a graphical indicator that a significant network failure has occurred.

If an interface fails on device R, as shown in Figure 31, APA emits the following two alarms:

- Interface Down R IfIndex 1
- Node Down N

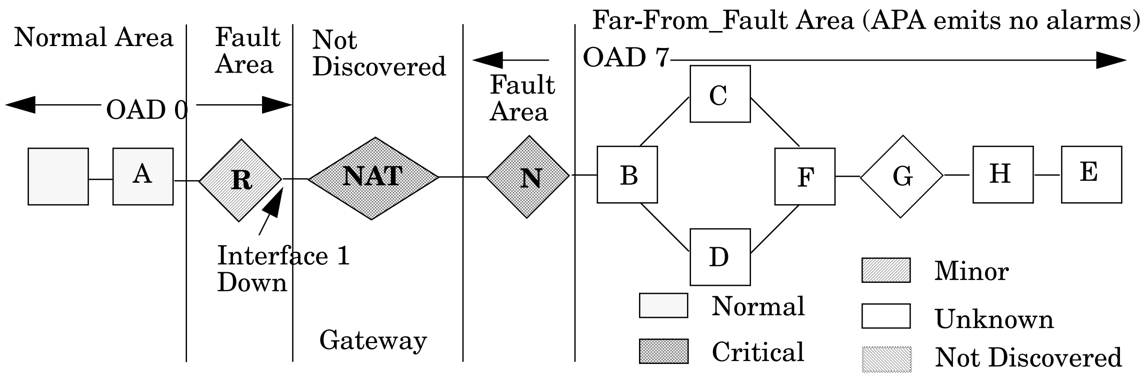
APA sets both interface 1 and device N to a critical status and sets device R to a minor status.

Figure 31 Interface 1 on Device R Fails



In the next example, Figure 32, the NAT or Gateway device is handled like the egress device because both the NextHop and Gateway devices are identified in the `dupip.conf` file.

Figure 32 APA Behavior with NextHop and Gateway Devices Configured



In Figure 32, interface 1 on device R fails, causing an entire OAD area to be inaccessible. This down interface normally connects to the NAT device in OAD 0. In this example, APA generates the following three alarms:

- Interface Down R IfIndex 1
- Node Down NAT
- Node Down N

APA analyzes NextHop and Gateway device failures more quickly than other devices and polls devices located in fault areas more quickly than devices located in Far-From-Fault areas.

Understand OAD View Status Information

The Active Problem Analyzer polls OAD addresses and reports device status information in the OAD view. See Chapter 4, Active Problem Analyzer for more information.

You can identify OAD alarms in the Alarm Browser. The alarms source contains an @ symbol followed by the name of the Overlapping Address Domain. See the *trapd.conf* reference page for more information. Details about how to access reference pages are in the online help.

Data Collections and Thresholds

You can enable Extended Topology to collect SNMP data for nodes configured in an OAD. This allows you to do the following:

- Collect MIB data from network nodes automatically at regularly scheduled intervals.
- Store the collected MIB data in a file.
- Set threshold monitoring on critical devices.

To enable Extended Topology to support SNMP data collection, use the following procedure:

1. Edit the following file:
   - *UNIX*:

     ```
     $OV_LRF/snmpCollect.lrf
     ```

   - *Windows*:

     ```
     %OV_LRF%\snmpCollect.lrf
     ```

2. Add a **-z** option between the two colons as shown by the bold text in the example below:

   ```
   OVs_YES_START:pmd,ovwdb,ovtopmd:-z:OVs_WELL_BEHAVED:20:PAUSE
   ```

3. As Administrator or root, run the following command:
   - *UNIX*:

     ```
     ovaddobj $OV_LRF/snmpCollect.lrf
     ```

   - *Windows*:

     ```
     ovaddobj %OV_LRF%\snmpCollect.lrf
     ```

4. As Administrator or root, stop and restart the `snmpCollect` process. See Stop and Restart Processes on page 17 for more information.

You cannot configure performance, availability, exception, or inventory reports for nodes configured in an OAD.

See *Managing your Network with Network Node Manager* for more information about Data Collection and Thresholds.

Glossary

A

ABR

Area Border Router. A router with an interface in more than one OSPF area.

Area

OSPF divides contiguous networks and hosts into a number of areas using Area Border Routers.

ARP

Address Resolution Protocol. A protocol that maps IP addresses to Ethernet addresses.

ATM

Asynchronous Transfer Mode. A high-speed connection-oriented switching technology that uses 53-byte cells (packets) to simultaneously transmit different types of data, including video and voice.

C

Community String

A plain text password used to authenticate SNMP queries to SNMP agents.

D

Dual Stack

A router or host that is configured for both IPv4 and IPv6.

Dynamic Views

Describes the family of browser-based views whose content is created as a result of choices you make when you launch the view, and which continue to provide the most current status information available.

H

Home Base

The main launching point for Dynamic Views.

HSRP

Hot Standby Routing Protocol. A proprietary Cisco protocol that provides backup to a router when it fails.

I

ILMI

Interim Local Management Interface. An independent industry standard used for configuring ATM interfaces.

IPv4-Compatible Addresses

Dual-stacked nodes understand both IPv4 and IPv6 addresses and use IPv4-compatible addresses to tunnel IPv6 packets through IPv4 routers. Ipv4-compatible addresses have their 96 high-order bits set to zero and 32 low-order bits set to an IPv4 address. Using this technique, dual-stacked nodes can use the same address for IPv4 and IPv6 packets.

IPv6 Global-Scoped Addresses

These addresses, normally referred to as aggregatable global unicast addresses, refer to IPv6 addresses that begin with 001. At some future time, IPv6 global-scoped addresses may include other currently unassigned unicast IPv6 prefixes.

IPv6 Link-Local Scoped Addresses

These addresses are only used to connect neighboring nodes. Link-local addresses always have the prefix 1111 1110 10 in the first ten bits. Almost all IPv6 addresses will have a link-local address.

IPv6 Prefix Group

A prefix group is similar to a subnet in IPv4. A prefix group has a specific scope: site-local scoped addresses or global-scoped addresses. There are no prefix groups for link-local scoped addresses.

IPv6 Site-Local Scoped Addresses

These addresses can only be used within an isolated internet. Site-local scoped addresses always have the prefix 1111 1110 11 in the first ten bits.

M

MIB

Management Information Base. In the context of managed devices, a collection of objects that can be accessed via a network management protocol.

N

Neighbors

Two devices that have directly connected interfaces.

O

OAD

Overlapping Address Domains. The OAD denotes a set of unique, private IP addresses. The OAD view is useful when monitoring several OADs that contain overlapping private IP addresses.

OSI Model

Open Systems Interconnect Model. A network design framework established by the ISO (International Standards Organization) to provide equipment from different vendors to be able to communicate.

OSPF

Open Shortest Path First protocol. A protocol that routers use to communicate between themselves. OSPF has the ability to configure topologies and adapt to changes in the Internet.

P

Port Aggregation

A method of connecting two Ethernet switches with two or more cables.

S

SNMP

Simple Network Management Protocol. A standard protocol used to manage TCP/IP networks.

T

Threshold

A preset, calculated, or MIB instance value used in NNM data collection. NNM can generate an alarm when a threshold violation occurs.

Tunneling

Enabling network communication between two IPv6 networks by forwarding IPv6 packets across an IPv4 network.

V

VCI

Virtual Channel Identifier. The address or label of an ATM Virtual Circuit.

VLAN

Virtual LAN. The creation of multiple logical networks within a single physical network or switching device. Each VLAN creates a new broadcast domain.

VPI

Virtual Path Identifier. The address of an ATM virtual path.

Index