Installation, User and Administrator Guide

Nimbra Vision

Release 5.6

© Copyright 2003-2010 by Net Insight AB, Sweden. All rights reserved. This document may not be reproduced in whole or in part without the expressed written permission of Net Insight AB.

The specifications and information in this document are provided "as is" and are subject to change without notice. All statements, information, and recommendations in this document are provided without warranty of any kind, expressed or implied, including without limitation any warranty concerning the accuracy, adequacy, or completeness of such specifications and information or the result to be obtained from using such specifications or information. Net Insight AB shall not be responsible for any claims attributable to errors, omissions, or other inaccuracies in the specifications or information in this document, and in no event shall Net Insight AB be liable for direct, indirect, special, consequential or incidental damages arising out of the use or inability to use this document.

Net Insight and Nimbra are trademarks of Net Insight AB, Sweden. All other trademarks are the property of their respective owners.

Net Insight AB
Box 42093
SE-126 14 Stockholm
Sweden
Phone: +46 8 685 04 00
Fax: +46 8 685 04 20
E-mail: info@netinsight.net

May 2010

Stockholm, Sweden

(NID2808 G3)

© 2003-2010 Net Insight AB, All Rights Reserved

Contents

© 2003-2010 Net Insight AB, All Rights Reserved

© 2003-2010 Net Insight AB, All Rights Reserved

© 2003-2010 Net Insight AB, All Rights Reserved

About This Manual

Overview

This manual contains information on how to use the Nimbra Vision Network Manager software. Nimbra Vision is an application for managing a network comprising of Nimbra network elements, such as Nimbra One, Nimbra 340, Nimbra 360, Nimbra 680 and Nimbra 688.

In addition to this manual, the *Web NMS User Guide* and *Web NMS Administration Guide* should be consulted.

Intended Reader

This manual is intended for network managers and administrators in operation and maintenance of network elements, that is using Nimbra Vision for network management. The manual is also intended for administrators of the Nimbra Vision itself.

Organization of Content

The contents of this manual is organized as follows:

- **About this Manual** includes information on how to use this manual.
- **Installation Guide** includes information on how to install and upgrade the Nimbra Vision.
- **User's Guide** described the Nimbra Vision additions to the Web NMS software, and is intended for the day-to-day user of Nimbra Vision.
- **Administrator's Guide** descries the Nimbra Vision specific procedures for administrating the Nimbra Vision.

Conventions in this Manual

To enhance the readability of this manual, sections of special importance or interest are marked as follows:

Information of Specific Importance

Note! *Information for proper function of the software is formatted like this paragraph.*

© 2003-2010 Net Insight AB, All Rights Reserved

Warning! *Especially important information for proper function is formatted like this paragraph.*

File Contents

Contents of files of console output are formatted as the next paragraph.

```
This is an example of a text file.
```

© 2003-2010 Net Insight AB, All Rights Reserved

Installation Guide

Requirements

This topic describes the requirements for the Nimbra Vision server and client software. Although there are no hardware requirements specified on the Applet and Java Web Start client, it is suggested that similar processor and memory requirements should be considered as for the Windows application client.

General

The host name of the server must be a valid Internet DNS name. Windows system accepts underscore ("_") as character in the computer's name, but this character is not valid in a DNS name.

Server Requirements

The server includes all the necessary software packages, such as MySQL database, Apache web server, Tomcat servlet engine and Java runtime environment. The specified server requirements should be considered as minimum requirements.

It is not recommended to run client and server on the same host when minimum requirements are just met.

Platform	Hardware minimum requirements
Windows XP (32-bit) Windows Server 2003 (32-bit)	Processor type: Pentium III Processor speed: 733 MHz Memory (RAM): 256 MB Hard disk: 600 MB

© 2003-2010 Net Insight AB, All Rights Reserved

Client Requirements

Application Client

Platform	Browser (for on-line help)	Hardware minimum requirements
Windows XP, Windows Server 2003 (32-bit)	Internet Explorer 6.0 Firefox	Processor type: Pentium III Processor speed: 733 MHz Memory (RAM): 256 MB Hard disk: 200 MB

Applet Client

Browser	Browser plug-in
Internet Explorer 6.0, 7.0, 8.0 Firefox 2.x, 3.x	Java 6 plug-in

Java Web Start Client

Browser	Browser plug-in	Java	Java Web Start
Internet Explorer 6.0, 7.0, 8.0 Firefox 2.x, 3.x	Java 6 plug-in	Java 6	Java 6

HTML Client

Web browsers must be have *cookies* and *JavaScript* enabled, and must *have pop-up blocking* disabled for pop-ups from the Nimbra Vision server. See your web browser documentation.

Browser
Internet Explorer 6.0, 7.0, 8.0 Firefox 2.x, 3.x

Nimbra Network Elements

NimOS is the system software running on the Nimbra network elements.

NimOS
GX4.1 or later

Installing the Server

To install the Nimbra Vision server, run the installation program `server.exe` and follow the instructions.

To complete the installation, you need to do configure Nimbra Vision and the Nimbra network elements. See Before You Start in the Administrator's Guide for details.

© 2003-2010 Net Insight AB, All Rights Reserved

Note! *Always install the software in an empty folder. Do not install the software in the same folder as a previous release. To upgrade, see Upgrading.*

You can have multiple installations of Nimbra Vision, but can only run one at the time.

The installation program will install the server, the Application Client, and necessary applications used by Nimbra Vision. The installed applications are MySQL database, Apache web server, Tomcat servlet engine and Java run-time environment. The installation of these applications will all be within the Nimbra Vision installation folder and will not affect the installation and configuration of any already installed applications.

Note! *Nimbra Vision comes with two pre-configure users,* `root` *with password* `public`*, and* `guest` *with password* `guest`*. The root* `user` *has full permissions, while the* `guest` *user has limited permissions.*

Installing the License Key

A license key is a file with information that grants you to run Nimbra Vision. The license key may contain a time limit on the license (typically for an evaluation license), and a configuration of different features, e.g. the number of managed nodes, redundant headends etc.

To be able to run the Nimbra Vision server application, a license key is required. If a license key has not been installed when Nimbra Vision starts, it will ask for you to enter the license key file after accepting the license agreement.

It can sometimes be necessary to install a license key on command. Some cases would be if:

- Upgrading from evaluation license to full license
- Upgrading the license to a new number of managed nodes
- Upgrade the license to include a new feature
- Start Nimbra Vision as a Windows service

To install a license key on command, navigate to *<NMS_Home>*`\bin\admintools` and run `license.bat`.

It is possible to install a new license key while the Nimbra Vision server is running. The Nimbra Vision server checks every two minutes to see if a new license key has been installed.

Installing the Server as a Windows Service

You can install the Nimbra Vision server and Apache server as a Windows Service. On installation as Windows Service, Nimbra Vision starts when the machine boots up along with other services such as Telnet, FTP etc. This topic will guide you in installing and uninstalling Nimbra Vision as a Windows Service.

Installing Windows Service

1. Run *<NMS_Home>*`\bin\adminstools\InstallNTService.bat`. The two services *NimbraVision-Service* (`JavaService.exe`) and *NimbraVision-Apache* (`Apache.exe`) will be installed, but not started.

© 2003-2010 Net Insight AB, All Rights Reserved

2. The Windows Services are now installed, and will be started next time the computer boots.
3. You can start the service *NimbraVision-Service* from the **Services** in Windows **Control Panel**. It will in its turn start the service *NimbraVision-Apache*. You can also start the service from a command prompt. Run:

```
NET START NimbraVision-Service.
```

4. If you have setup the server with Server Failover, then you must ensure that the MySQL database is started before the *NimbraVision-Server* service is started. See Server Failover.

When Nimbra Vision is started as a Windows service, an error message will be logged into *<NMS_home>*`\logs\stderr.txt`. It does not affect the performance. See below for details.

Disabling Starting of Apache by the Nimbra Vision Server

When Nimbra Vision is installed as services, Windows will start Apache, which normally is started by the Nimbra Vision server. When Nimbra Vision server running as a Windows service tries to start Apache, it will now fail doing so and log an error into *<NMS_home>*`\logs\stderr.txt`. This is because the service would already have been started. The failure in starting the Apache will not affect the service, but will log an error message. If you want to avoid the error message, you must disable the starting of Apache by the Nimbra Vision server. However, you will in this case not be able to start Nimbra Vision except as a service.

To disable starting of Apache, use a text editor (e.g. **Notepad**), edit the files *<NMS_Home>*`\conf\NmsProcessesBE.conf` and *<NMS_Home>*`\conf\NmsProcessesFE.conf`. In these two files, comment the two lines that start Apache. You do this by inserting a '#' in the beginning of the lines. The lines shall look like below when edited:

```
#java com.adventnet.nms.util.RunApacheExeModule ARGS
#PROCESS             com.adventnet.nms.util.RunApacheExeModule
#ARGS                StartApache -d apache -f conf/httpd.conf
```

The recommendation is to not change the configuration file. At the cost of an error message, you will be able to start the Nimbra Vision server as an application or as a Windows Service.

Note! *The license key file must be installed before you start the Nimbra Vision server as a Windows service. See Installing the License Key on details on how to install a license key file.*

Uninstall Windows Service

1. Shut down the services from the **Services** function in Windows **Control Panel**, or from a command prompt, run:

```
NET STOP NimbraVision-Service
```

2. Run *<NMS_Home>*`\bin\admintools\UninstallNTService.bat`.

© 2003-2010 Net Insight AB, All Rights Reserved

3. The service is now uninstalled.
4. You must restore the configuration file `NmsProcessesFE.conf` if you changed it when you installed the service. Remove the '#' from the lines that you commented when you installed the service. See above.
5. If you were using server failover, then you must also uninstall the MySQL service. See Server Failover.

See Also

Installing the License Key

Server Failover

Installing the Application Client

To install the Nimbra Vision client, run the installation program `client.exe` and follow the instructions.

The program will install the Application Client and the Java run-time environment. The installation of the Java run-time environment will be within the Nimbra Vision client installation folder, and will not affect any other Java installations.

***Note!** The Application Client software is already installed together with the server software. You do not have to install the client software on the server as a separate installation.*

Installing the Applet or Java Web Start Client

The Applet Client and Java Web Start Client are downloaded from the server when it is started. To run the applet or Web Start client, the Java Runtime must already be installed. See Client Requirements.

In addition, the following policy permissions musts be granted on the client host:

```
grant {
   …
  permission java.util.PropertyPermission "user.dir", "read";
  permission java.io.FilePermission "${java.home}\\lib\\rt.jar", "read";
  permission java.lang.RuntimePermission "queuePrintJob";
  permission java.util.PropertyPermission "reload.path", "read";
  permission java.net.SocketPermission "*", "connect";
  permission java.awt.AWTPermission "setAppletStub";
   …
};
```

It is recommended to use `policytool` provided with the Java Runtime to add or edit the permissions. The `policytool` is found as *JAVA_HOME*`\bin\policytool.exe`.

The policy can be set per user, or for the system. On Windows XP, the policy per user is saved in the file `%USERPROFILE%\.java.policy`. For the user Administrator, this typically is `C:\Documents and Settings\Administrator\.java.policy`. On Linux, it is saved in `~/.java.policy`.

If there is no per-user policy file, it might be easier to create it using a text editor and add the text as described above, but exclude the ellipsis ("...").

© 2003-2010 Net Insight AB, All Rights Reserved

To grant for the system, you can set the permissions in the system policy file `JAVA_HOME/lib/security/java.policy` on your client host.

Installing the Standalone Front-End Server

To install the Nimbra Vision FE server, run the installation program `fe_server.exe` and follow the instructions.

The Standalone FE server should not run on the same host as the Nimbra Vision server. (The Nimbra Vision server is already running a FE server).

The FE server must be connected to the BE (back-end) server located in the Nimbra Vision server. You must enable access from the FE to the database. On the Nimbra Vision server, start the database and grant permissions as described below. The database is running if you have started the Nimbra Vision server.

You can install a FE server after you have Nimbra Vision server running, without shutting down the server.

1. Check the text file `<NMS_Home>\conf\database_params.conf` for the password that is used by Nimbra Vision to connect to the database. Start a command prompt, and navigate to `<NMS_Home>\mysql\bin`, and start the `mysql` client, where *password* is the password:

```
mysql -uroot -ppassword
```

2. The FE server must be able to login to the server database. It will use the login user root. Execute the statement, where *fe-sever* is the name or IP address of the host where the FE server shall run, and *password* is the password used for accessing the database, i.e. same password as above:

```
mysql> GRANT ALL ON *.*
     -> TO 'root'@'fe-server'
     -> IDENTIFIED BY 'password';
```

3. Exit the `mysql` client:

```
mysql> EXIT
```

4. On the host where the FE is installed, edit the file `setEnv.bat` using text editor (e.g. Notepad). In the beginning of the file, assign the hostname or IP address of the Nimbra Vision back-end server to `BE_HOST`. All client hosts must know the hostname. If not, then the clients may fail when connecting to this FE server in case of FE failover.

```
BE_HOST=nvserver
```

5. The hosts file should be the same as on the BE server. See installation of server above.
6. You are now ready to start the FE server. To start, use the installed shortcut on the Start menu, or run `<NMS_Home>\bin\startnmsFE.bat`.

Upgrading

- Normal upgrading procedure
- Upgrading when running server failover

© 2003-2010 Net Insight AB, All Rights Reserved

- Upgrading when running external database

Normal upgrading procedure

If you are using Server Failover, see upgrading procedure below.

You can upgrade from an earlier release on Nimbra Vision. Upgrade is supported from version 5.0. To upgrade:

1. Install the server (see above). Do not install it over an existing installation. Note that because the server is not installed in the same folder as the already existing earlier release, the installation will not affect any existing installation.
2. If Nimbra Vision was already running, stop the Nimbra Vision server application. If it is running as a Windows service, stop the service. Make sure that the Nimbra Vision server application is not running, including the MySQL database.
3. Optionally, install the license to avoid the upgrade program to ask for this when it started later. To install a license key on command, navigate to `<NMS_Home>\bin\admintools` and run `license.bat`. See Installing the License Key for details.
4. To import data from the Nimbra Vision that you are upgrading from, run the upgrade program. To do this, navigate to `<NMS_Home>\bin\admintools` of the newly installed Nimbra Vision and run `upgrade.bat`. The program will prompt for the location of the Nimbra Vision wherefrom the upgrade shall be made. The program will:
   - Copy the database,
   - Convert the database,
   - Copy all user settings and folders,
   - Copy necessary configuration files. The original files will be saved in the folder `<NMS_Home>\orig`.
5. You can now start the newly installed and upgrade Nimbra Vision server. If the server is to be run as a Windows service, you must first uninstall the previous service before installing the new Windows service. You can do this while the Nimbra Vision server application is running if you did not start the server a Windows service. See Installing the NMS Server as a Windows Service.
6. In Nimbra Vision 5.5, the definitions of users, groups, and permitted operations, have been remodeled, allowing for fine-grained authorization. When upgrading from release before 5.5 all group assignments for all users except `root` are removed. All groups are removed and replaced with default groups, and all operations are removed and replaced with default operations. For `root`, a default assignment of groups is made. All users except `root` are therefore disabled. You will have to login as `root`, assign applicable groups for all users, and re-enable the users.
7. Done.

Note the following:

- The Reporter Reports are not copied by the upgrade program `upgrade.bat`. (See Performance Reports in the User's Guide for details about these reports). If Reporter Reports has been generated (reports on DTM interface usage) and you would like to access these

© 2003-2010 Net Insight AB, All Rights Reserved

from this version of Nimbra Vision, you need to move or copy its folder manually: copy or move the folder `reports\dtmIfReports` with all its contents. You can do this at a later time, even when the server is running. See the Generate Reports policy in the User's Guide for its function.

- The database backups and node configuration backups are not copied by the `upgrade.bat`. These are all stored in the `backup` folder.
- You cannot use the database backups from the previous releases without converting the database. You can use the upgrade program `upgrade.bat` to convert the database.

Upgrading when running server failover

The upgrade procedure when you are using server failover (redundant servers) is not much different from a standard upgrade. Please also read Normal upgrade procedure above for details.

To upgrade:

1. On the standby server: Shut down the Nimbra Vision application.
2. On the standby server: Shut down the database. See Stopping the Server in Administrator's Guide.
3. On the primary server: Install the server software.
4. On the primary server: Optionally install the license to avoid the server to ask for this when it is later started.
5. On the primary server: Shut down the Nimbra Vision application.
6. On the primary server: Shut down the database. See Stopping the Server in Administrator's Guide.
7. On the primary server: In the new installation, run the upgrade program: navigate to *<NMS_Home>*`\bin\admintools` and run `upgrade.bat`. This will upgrade the primary server and copy the database.
8. On the primary server: edit the file *<NMS_Home>*`\mysql\my.ini` which was copied by `upgrade.bat`. Ensure the following:
   - `wait_timeout=2073600` is removed.
   - `slave_net_timeout=180` exists.
9. On the primary server: Start the new database. See Starting the Server in Administrator's Guide.
10. On the primary server: Start the Nimbra Vision Server. The primary Nimbra Vision server is now up and running and can be used.
11. On the standby server: Install the server software.
12. On the standby server: copy the file *<NMS_Home>*`\mysql\my.ini` from the old to the new installation, overwriting the file in the new installation. This is the MySQL configuration file.
13. On the standby server: edit the file *<NMS_Home>*`\mysql\my.ini`. Ensure the following:
    - `wait_timeout=2073600` is removed.
    - `slave_net_timeout=180` exists.

© 2003-2010 Net Insight AB, All Rights Reserved

14. On the standby server: Edit the file `<NMS_Home>\conf\database_params.conf`. Update the url entry: replace `localhost` with the hostname or IP address of the server.
15. On the standby server: Copy the folder `<NMS_Home>\mysql\data` from the old to the new installation, overwriting the files in the new installation. This is the MySQL database content.
16. On the standby server: Start the new database. See Starting the Server in Administrator's Guide.
17. On the standby server: Verify that the database is replicating. See Verifying Database Replication in Administrator's Guide.
18. On the primary server: verify that the database is replicating.
19. On the standby server: start the Nimbra Vision server. The server is now starting in standby.
20. Done.

Upgrading when running external database

This section describes how to upgrade when the database is started outside of the Nimbra Vision installation. The database could either be running on the same host as Nimbra Vision, or on a remote host.

If the database is running outside of Nimbra Vision, then the upgrade program will detect this running database, and upgrade its format. The upgrade program will not make a copy of the running database. You should make a backup of the database to be able to revert.

Uninstallation

Uninstalling the Server

Before uninstalling the Nimbra Vision server application, the application must be shutdown. If the Nimbra Vision server application is running as an Windows service, you must shutdown and remove the service before uninstalling the software. See Installing the NMS Server as an Windows Service for details on removing the service.

After the Nimbra Vision server application has been shutdown, and the Windows service removed, run the uninstallation program as:

Start | Programs | Nimbra Vision *version* | Uninstall Nimbra Vision.

-or-

Uninstall from the **Add/Remove Programs** function in Windows.

If you have made multiple installations of the same version of Nimbra Vision, then the Start menu is referring to the latest installed Nimbra Vision. The same for Add/Remove Program. You can uninstall a specific installation by running **Uninst.exe** in the Nimbra Vision root folder.

Uninstalling the Client

To uninstall the client, run the uninstallation program as:

© 2003-2010 Net Insight AB, All Rights Reserved

Start | Programs | Nimbra Vision Client ***version*** **| Uninstall Nimbra Vision**.

-or-

Uninstall from the **Add/Remove Programs** function in Windows.

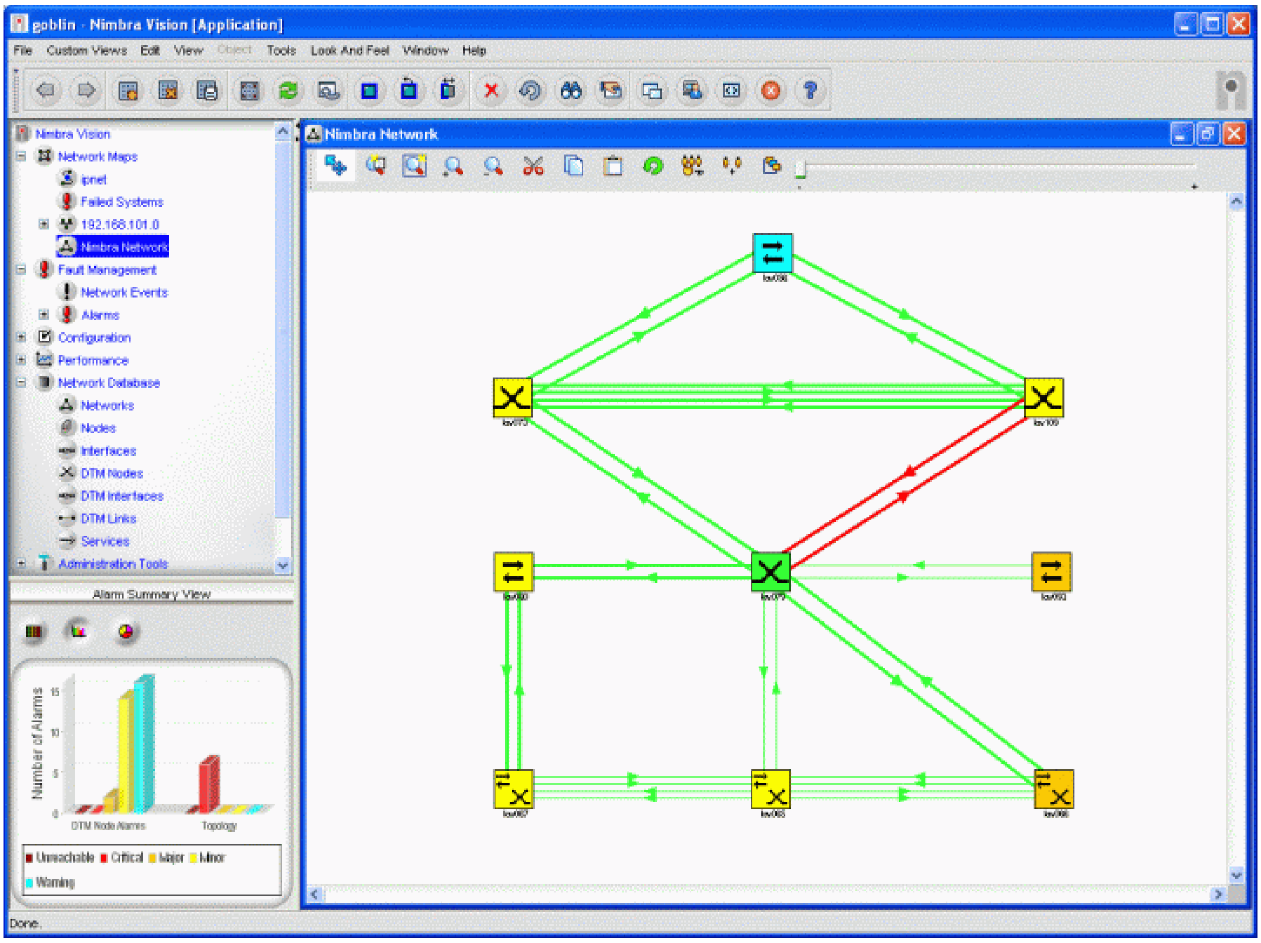
If you have made multiple installations of the same version of Nimbra Vision application client, then the Start menu is referring to the latest installed application client. The same for Add/Remove Program. You can uninstall a specific installation by running **Uninst.exe** in the Nimbra Vision application client root folder.

© 2003-2010 Net Insight AB, All Rights Reserved

User's Guide

Introduction

Nimbra Vision is an application for managing Nimbra networks. It consists of maps, a network database, an event and alarm database and a number of applications for managing the nodes and the network itself. The image below shows a typical window when the client has been started.



The Nimbra Vision client window, with Tree for navigation, alarm summary panel and a Nimbra Network map.

The main window is divided into a number of sections:

- On the top of the window is a **menu bar** with menus.

© 2003-2010 Net Insight AB, All Rights Reserved

- Below the menu bar is a **tool bar**. The toolbar contains commonly used commands for the frame that is currently active.
- At the left side of the window is the **tree** where you can navigate between different panels. By selecting a node in the tree, the corresponding frame will be displayed.
- Below the tree is the **alarm count panel**. It shows the number of alarms per severity in different categories. You can quickly access the alarm frame by double-clicking in the panel. The alarm frame will show a list of alarms corresponding to the double-clicked item. The following categories are used by Nimbra Vision, but note that the user could create new categories:
    - **DTM Node Alarms**. Alarms in this category are raised and cleared in the nodes, and the information is forwarded to Nimbra Vision. The nodes detect these alarms.
    - **Topology**. Alarms related to the network topology. Includes alarms raised when Nimbra Vision fails to connect to a node or one of its IP addresses, and alarm when Nimbra Vision detects change in DTM topology. Nimbra Vision detects these alarms.
- To the right of the tree, and also using the major part of the window, is the **display panel**. The display panel contains all the frames that are displayed, and it works as a desktop. You can change the size, maximize, minimize and move the different frames within the display panel. It is also possible to detach a frame, making it a separate window.
- The window title contains the hostname or IP address of the Nimbra Vision server to which the client is connected. If the front-end server and back-end server are not the same, then the hostnames of both are displayed in that order separated by a slash.

Working with the Tree

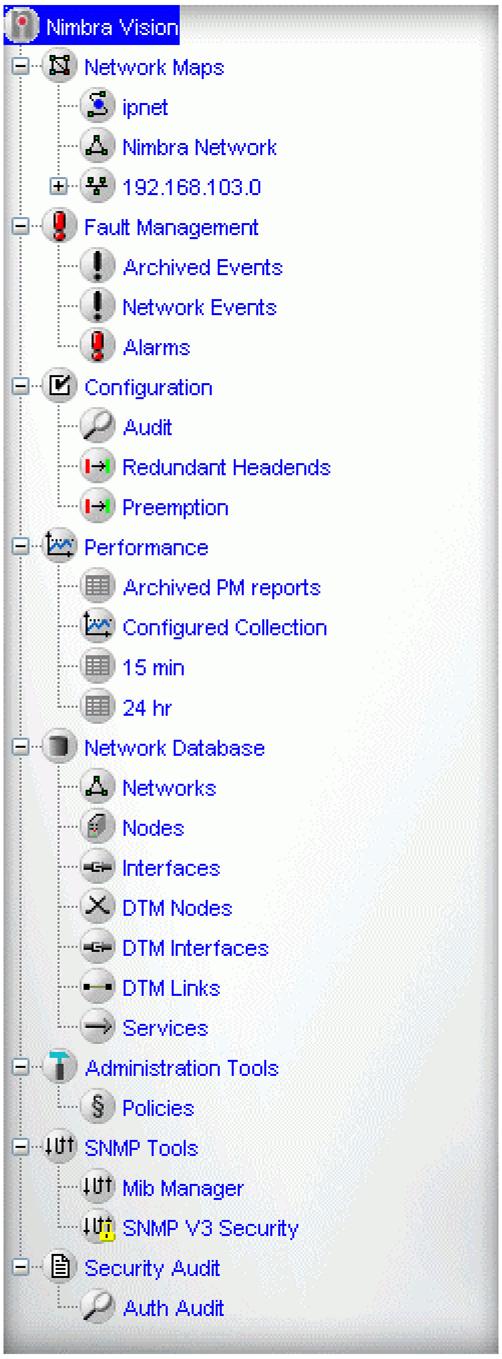
Navigating the Tree

The tree is located on the left side of the main window. It consists of a number of nodes that represents different panels that are shown in the main part (display panel) of the main window. This topic describes the nodes in the tree.

In the tree you can:

- Expand and collapse nodes
- Select a node, which opens the corresponding panel in the display panel.
- Modify the tree by adding and removing nodes. Custom Views are typically such nodes.

© 2003-2010 Net Insight AB, All Rights Reserved



The Tree for navigating to different panels.

Node	Description
ipnet	Map with IP networks as managed objects, i.e. with symbols representing the different sub-networks. Double-clicking on a network symbol opens the map corresponding to the IP network. The map also displays routers between the different IP networks. The map gives a complete view of the management network.
Nimbra Network	The DTM Map contains all the Nimbra nodes, and links.

© 2003-2010 Net Insight AB, All Rights Reserved

Node	Description
192.168.103.0	Each IP network with all its containing nodes. There is one map per sub-network. Sub-maps to this map contain all failed nodes, defined as nodes with status severity *Critical* or *Major*.
Fault Management	Contains panel related to fault management; events and alarms.
Network Events	List with all the network events. Network events are results of received SNMP notifications, or as generated by Nimbra Vision. See Events for details about the events.
Alarms	List with alarms. The list can also be opened from the Alarm Count Panel. Its content depends on the current filtering. See Alarms for details about the alarms.
Archived Events	Search and view archived events. See Archived Events.
Audit	Audit log, logs results from configuration changes of the nodes, e.g. backup, redundant headend switch-over, etc.
Redundant Headends	Configuration of redundant head-end functionality. See Redundant Headends for details.
Preemption	Configuration of pre-emption functionality. See About Pre-emption for details.
Performance	Contains panels with performance related information.
Configured Collection	Manages and displays collection of statistics. Allows plotting of data. One type of collected data could be the usage of the DTM interfaces.
15 min	15-minute performance data according to G.826 on trunk interfaces and access interfaces. The information is received from the nodes as events every 15 minutes, and can also be seen in the **Network Events**. The custom view shows Performance Monitoring events. • IS, indicates if the performance report is suspect, i.e. it may not contain correct data. • ZS, zero suppression counter, number of consecutive measurement periods where no report has been generated because the report contains all zeroes. • ES, errored seconds • SES, severely errored seconds • BBE, background block errors • UAS, unavailable seconds • SS, slip seconds See Event: Performance_Monitor_Event for details about performance data.
24 hr	24-hour performance reports similar to the 15-minute reports. See **15 min** for details. See Event: Performance_Monitor_Event for details about performance data.
Archived PM Reports	Search and view archived G.826 performance monitoring reports. See Archived PM Reports.

© 2003-2010 Net Insight AB, All Rights Reserved

Node	Description
Network Database	This node contains different views of the Network Database. The different views does also display different properties of the managed objects. The following custom views exists in a new system by default: • Networks: IP networks, i.e. **isNetwork**=*true*. • Nodes: All nodes, i.e. **isNode**=*true*. • Interfaces: All the interfaces, i.e. **isInterface**=*true*. • DTM Nodes: All DTM nodes, i.e. **className**=*DtmNode*. • DTM Interfaces: All DTM interfaces, i.e. **classname**=*DtmInterface*. • DTM Links: All DTM by-pass link connections, i.e. **classname**=*DtmLink*. • Services: All ITS and ETS services, i.e. **classname**=*DtmService*
Administration Tools	Tools and panes for administration.
Policies	The word *policy*, in general, refers to a plan of action. Here, in this context, it refers to executing a task or set of tasks at a specified time based on a set of specified conditions. Policies are tasks that are executed by Nimbra Vision at a system level, at a specified point of time. Policies are used to control a variety of network activities like automated backups, generating reports, cleaning up database tables, escalating alarms, etc.
SNMP Tools	Tools and panels for working with and configure SNMP.
Mib Manager	An SNMP MIB browser.
SNMPv3 Security	Configure and update the USM tables for SNMPv3 security. This includes the user name and password, security level etc.
Security Audit	Audit trails for user executed operations in Nimbra Vision.

Working with Maps

Nimbra Network Map

This topic covers the following sub-topics:

- Symbols
- Colors
- Links
- Map Operations

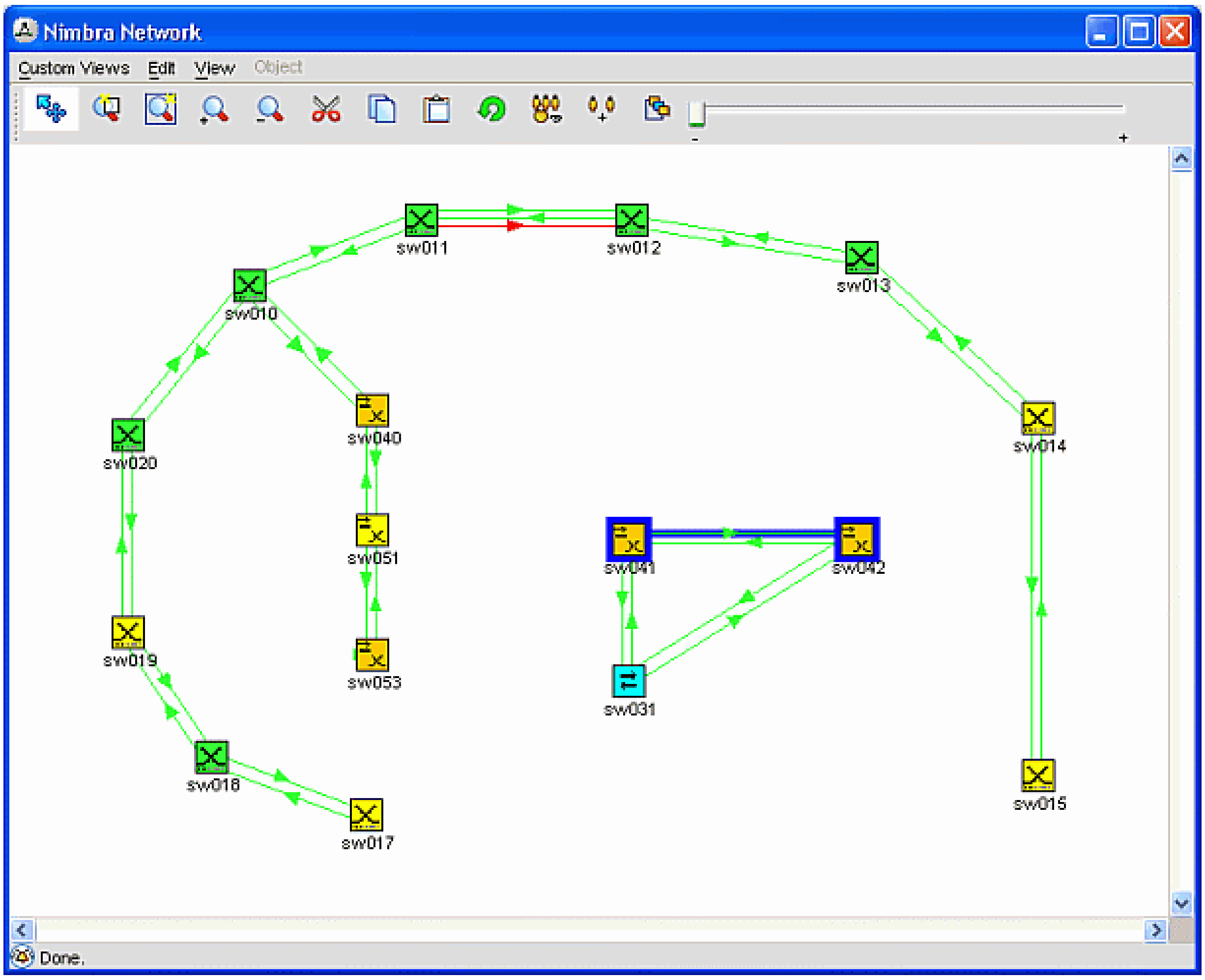
The map *Nimbra Network* is a graphical representation of the Nimbra network. The map shows the Nimbra nodes, such as switches and access nodes, and their topology with the links. Note that the map does not show the same information as the IP maps, although it may show the same nodes. The IP map shows the network from an IP network view.

The Nimbra nodes are represented by symbols, where the color represents the highest alarm severity on the node.

© 2003-2010 Net Insight AB, All Rights Reserved

The links are represented by arrowed links, where the color represents the link status, and the thickness represents the link capacity.

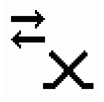
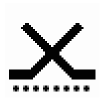
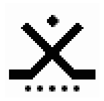
The default DTM map, **Nimbra Network**, shows all discovered Nimbra nodes.



Nimbra network map window with two nodes and a link highlighted (blue).

Symbols

The DTM nodes are represented by symbols. Different symbols represent different type of nodes.

Symbol	Type	Description
	NimbraOne	Nimbra One
	Nimbra290	Nimbra 290
	Nimbra291	Nimbra 291

© 2003-2010 Net Insight AB, All Rights Reserved

Symbol	Type	Description
⇄	Nimbra340 Nimbra340HD Nimbra360 Nimbra360Gold	Nimbra 340, Nimbra 340 HD, Nimbra 360 or Nimbra 360 Gold
✕	Nimbra680 Nimbra688	Nimbra 680 or Nimbra 688
⇄ JPEG 2000	Nimbra130/SD-SDI-codec Nimbra130/HD-SDI-codec Nimbra130/HD-SDI-decoder	Nimbra 130, with different software options.
✛	NimbraUnknown	All other Nimbra nodes.

Colors

The color of the symbol represents the status of the node. I.e. the highest severity of an active alarm, or a color representing unknown if the node is unmanaged.

Severity	Color Name	Color
Unreachable	Maroon	🟫
Critical	Red	🟥
Major	Orange	🟧
Minor	Yellow	🟨
Warning	Cyan	🟦
Clear	Green	🟩
Information	White	⬜
Unmanaged	Gray	▨

Links

The network links are also shown as links between the node symbols in the map. An arrow on the link shows the direction of the link. The color of the link reflects the link status. See Nimbra Link Properties for details about link status.

The thickness of the link represents the maximum transmit capacity of the link's originating interface.

It is possible to show an administrative name for the link in the map. The administrative name is a text sting that you can supply for easy identification of the link.

Map Operations

You can perform all the map operations on Nimbra maps as on other maps. See the Web NMS User Guide for description on map operations.

When a node or link is removed from the network, Nimbra Vision will detect that it cannot connect or reach the node any longer, and will raise an alarm. When you have verified that these objects really are removed from the network, you can delete these objects manually. This procedure will ensure that only removed objects are deleted, and not objects that cannot be reached due to network errors.

© 2003-2010 Net Insight AB, All Rights Reserved

Resizing Maps

When a map is saved in the database, it is saved with its current size. A map can only be saved if it not zoomed.

When a map is loaded into the map window, it is loaded with its saved size, and the symbols will always be placed at their saved positions. If the map would be larger than the window then the window would have scrollbars, and if the map would be smaller than the window then the map would take necessary space in the top-left corner but the background image would fill the window.

If you change the window size, then the map will be stretched accordingly, and all symbols are moved to their stretched positions. The new map size and the new symbol positions are only local to the client session. To permanently save the map with its new size and symbol positions, you must explicitly save the map.

A map is loaded when it is first opened, or when doing a refresh on the map. To do a refresh, choose menu **View | Refresh** or the toolbar button **Refresh**.

If the map size is according to your preferences but the window size is not, then correct the window size (which will stretch the map), and then refresh the map, which will load the saved map into the re-sized window.

If the window size is according to your preferences, but the map size does not fit the window as you like, then resize the window to stretch the map to the preferred map size, save the map, resize the window to its preferred window size, and finally refresh the map to load the saved map with its correct size into the window with its correct size.

If you want to have the whole map displayed in the window, then you can instead use the function Fit Map to Window It will stretch and resize the map and symbol positions that they all fit into the window. To fit the map to the window, choose menu **Custom View | Fit Map to Window** or the toolbar button **Fit Map to Window**.

Adding a Map

You can add a map from the client. This topic describes how to add a map that is already configured for DTM nodes.

Steps Involved in Adding a Map

You can add a map by following the steps given below:

1. Invoke the Add Map Property form using the menu **Custom View | Add New Map**, or the Add Map button.
2. Fill in the Map Properties
    a. The **Name** is name of the map in the database. This name has to be unique. Nimbra Vision will automatically append `.netmap` to the name.
    b. The **Label** is the text that will be displayed in the Tree.
3. Specify the match criteria to select what managed objects that will be included in the map. See Map Properties details on how to write a match criterion.
    a. Check **Create custom map** if you want the map to automatically be populated with symbols and links according to match criteria. Or uncheck **Create custom map** if you want to create an empty map. You will in this case have to add

© 2003-2010 Net Insight AB, All Rights Reserved

symbols and links to this map manually. You can do this by cut-and-paste from other maps.

   b. **Name** is the match criterion for selecting which managed objects that will populate the map. Knowing that the names of links are based on the node's names makes it easy to specify a criterion.

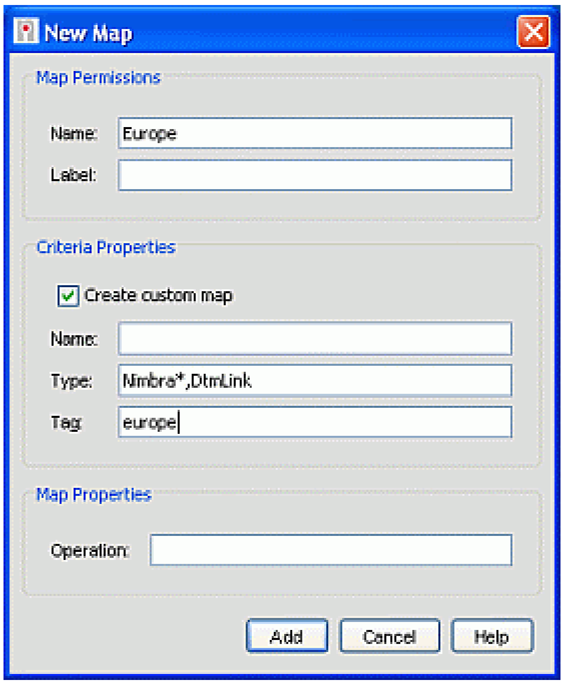
   c. **Type** is a match criterion for the type of managed object.

   d. **Tag** is a tag that you have assigned to managed objects.

4. If you want enable access permission constraints on the map, add an **Operation**. If the operation is empty, then all users will be able to access the map. If the operation is non-empty, then only users (including yourself) that has this operation granted will be able to access the map. To setup operations for different users and groups, use the Security Administration. See Edit Map Permissions to change the operation on a map after the map has been created.

5. Click **Save** to add the map. A node is added under the Network Maps node in the Tree representing the map.

6. After the map has been created you can modify the Criteria Properties to add additional criteria. To do this, open the Custom View properties for the map and add additional criteria. To open the Custom View properties, ensure that no objects are selected in the map and



Name: Europe
Label:

[✓] Create custom map
Name:
Type: Nimbra*,DtmLink
Tag: europe

Operation:

The New Map dialog.

See Also

Map Properties

Tagging Managed Objects

Edit Map Permissions

Writing Filter Criteria

Managing Operations in the Web NMS Administrator Guide

© 2003-2010 Net Insight AB, All Rights Reserved

Edit Map Permissions

All maps are shared among all users. To restrict access permissions on a map, an operation can be assigned to the map. Only users who are granted the operation will be able to access the map. If no operation is assigned to the map, then all users will be able to access the map. An operation can be assigned to the map when it is created, or using the Edit Map Permissions dialog.

To edit access permissions on an existing map, choose the menu **Custom View | Edit Map Permissions**. This opens a dialog that shows the operations assignment for all maps.

The Edit Map Operations dialog shows a table with all maps.

- **Map Label** is the label of the map, as seen in the tree
- **Map Name** is the unique name of the map in the database
- **Operation** is the operation string. Click in the cell to edit the value to change the operation, and click button **OK** to apply.

Note! *If the access permission is changed for a user, either by changing the operation assigned to a map, or by changing the permission granted for a user or user group, then the affected user must restart the client for the change to take effect.*

See Also

Managing Operations in the Web NMS Administrator Guide

Map Properties

To view and edit the details about a map, ensure that no symbols are selected in the map and choose menu **View | Symbol Properties**.

Details for the DTM Map

The details for the map are described in the table below.

Property	Description
name	The name of the map in the database. This must be a unique name among the maps.
label	The *display name* or *label* of the Map. This name will be displayed in the tree.
topology	The permissible layouts for the map. You list multiple layouts separated with commas. Some layouts are `grid`, `star`, `ring`, `flow`. For grid layout, by default all symbols will be resized automatically depending on number of symbols, map canvas size etc. This means that even if a symbol size is later modified using the symbol property editor, it will be resized to match the rest of the symbols. It is possible to add parameters to the grid layout so that it will ignore resizing symbols that are **anchored**. To do this, set: `grid(resizeAnchored=false)`
currenttopology	The topology that will be used per default. If no topology is given, the first topology in the **topology** will be used. If no topology is given in **topology**, then the *flow* layout will be used.
imageName	The background image. This is a file name relative the *<NMS_Home>*`\images` folder.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
autoplacement	The value is *true* if the layout will be applied, otherwise *false*. The default value is *true*.
menuname	A comma-separated list of panel-specific menu files for the map. For a DTM map, there are no dedicated menus.
groupname	Not used.
helpDoc	The URL for the help document that is to be invoked for help on the map.
mapSymbolRenderer	The renderer that paints the map symbols in the map canvas. Specify the full Java class name of the renderer. For the DTM map, the renderer used is: `se.netinsight.nv.nm.MapSymbolRendererImpl`. Other renderers that exist are `com.adventnet.nms.mapui.MapSymbolRendererImpl1`, `com.adventnet.nms.mapui.MapSymbolRendererImpl2` and `com.adventnet.nms.mapui.MapSymbolRendererImpl3`.
maplLinkRenderer	The renderer that paints the links on the map canvas. Specify the full Java class name of the renderer. For the DTM map, the renderer used is: `se.netinsight.nv.nm.DtmMapLinkRenderer.`
anchored	If the property is set to *true*, then the map becomes a non-editable map. It is not possible to move symbols on a non-editable map. Default is *false*.
treeIconFileName	The file name of the image icon that will be shown in the tree for the map. This is a file name relative the *`<NMS_Home>`* folder. For a DTM map, the file name is `images/nv_dtmnettreeicon.png`.
parentNode	The parent node in the tree for the map.

Match Criteria

To add match properties, click the **More…** button. Add matching criteria for the managed objects that will be represented in the map. You can match on any of the properties of the managed object.

For a managed object to be presented in the map:

- For the single criterion, the property has to match any of its rules. The rules can be specified in a comma-separated list. You can use the asterisk (`*`) as a wild card, both in the beginning, within or at the end of a match rule. If the property does not exist for a managed object, it is not considered as a match.
- For multiple criteria, all the single criteria must match.

For the DTM map, the matching criteria is as shown in the table below. This will match all the DTM nodes and links.

Property	Rules for the DTM Map
type	Nimbra*, DtmLink

To create a map that only includes a subset of the nodes, you have to supply additional criteria. You can use the **name** or **displayName** property for this. Knowing that the links name always is constructed from the name of the DTM interfaces, and that the DTM interface name is constructed from the Nimbra

© 2003-2010 Net Insight AB, All Rights Reserved

node name, we can setup a criteria that only selects the objects that has a certain sub-string in its name. The table below adds a criterion that will match:

The first criterion selects all the Nimbra nodes and links.

The second criterion restricts to managed objects which name ends with `.lab` (matches the Nimbra nodes), or which names includes the string `.lab-` (matches the links).

Property	Rules for a map with object name constrains
type	Nimbra*, DtmLink
name	*.lab, *.lab-*

See Also

Adding a Map

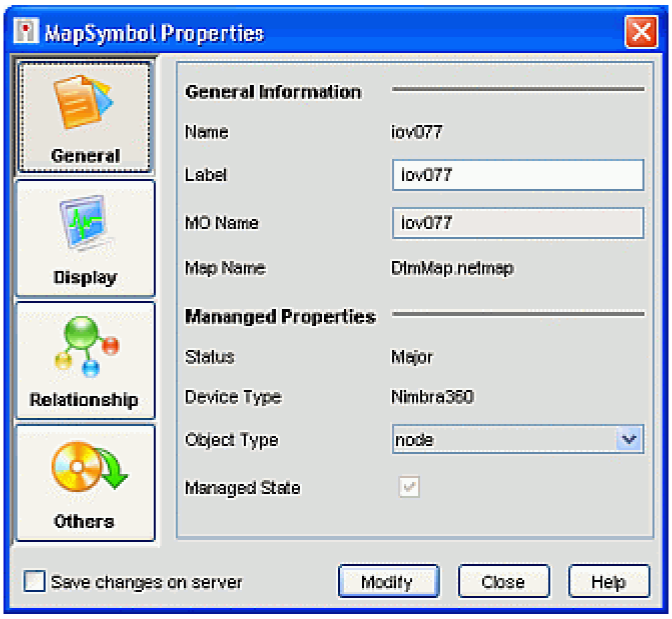
Writing Filter Criteria

Map and Device Details in the Web NMS User Guide

Accessing Symbols Details

The dialog Symbol Properties is a dialog that displays all the properties about a symbol in a map. The dialog is generic and its function is the same for all the symbols regardless of their type or class.

To open the Symbol Properties dialog, select a symbols object in a map, and choose the menu ***Object* | Symbol Properties**. If no symbol is selected, then the Map Properties are displayed.



General Information

Name iov077

Label iov077

MO Name iov077

Map Name DtmMap.netmap

Mananged Properties

Status Major

Device Type Nimbra360

Object Type node

Managed State ✓

☐ Save changes on server

The Map Symbol Properties dialog.

You can edit some of the properties. If you modify the properties, the database will be updated, but change will not propagate to the node. Generally, the dialog is not used for changing the information and the ability to save the edited data should be disabled for most users.

© 2003-2010 Net Insight AB, All Rights Reserved

See Also

Map Symbol Properties in the Web NMS User Guide

Highlight in Maps

Highlighting is a function that marks symbols in maps with a color. The purpose is to display the result of some operation in the map, e.g. tracing a channel. Some function can also use highlighted objects as inputs, such as creating a source route. Some operations highlight objects in different colors, where the color represents some property.

All symbols in all maps that are representing the highlighted object are marked. Successive highlight operations add or update (overwrite) highlighted objects. To clear highlighted objects, use the menu **Edit | Clear Highlight** or the toolbar button **Clear Highlight**.

Highlight is displayed in one of two modes:

- Show severity. In this mode, all highlighted symbols painted with a frame using the highlight color, and the symbols are painted with the color representing the status of the object. This mode is automatically selected if all objects are highlighted using the default blue color.
- Hide severity. In this mode, all the highlighted symbols are painted in the highlight color, and the remaining symbols are painted in gray. The severity of the status of the symbol is not displayed. This mode is automatically selected if any symbol is highlighted in another color than the default blue color.

You can toggle back and forth between the display modes Show severity and Hide severity. Use the menu **Edit | Toggle Highlight Mode** or use the toolbar button **Toggle Highlight Mode**.

To highlight objects:

- To manually highlight objects, select the symbols in a map that represents the objects, or select the managed objects in the Network Database, and choose menu **Edit| Highlight Selected Objects** or use toolbar button **Highlight Selected Objects**. The symbols are highlighted in the order that the symbols were initially selected.
- To highlight links based on the link utilization, see Highlight Link Utilization.
- To highlight a traced channel, see Trace Channel.
- To highlight a traced synchronization, see Trace Network Synchronization.
- To highlight a source route, see Source Route Editor.

Hierarchical Maps

Hierarchical maps is a way of building maps in hierarchies so that nodes in e.g. a sub-net, area, or on a site, can be represented by a single symbol in the map. This symbol can be opened or expanded to view the contents of the group.

There are two ways to make map hierarchies:

1. Group symbols into map groups
2. Add a map symbol linked to another map (sub-map)

© 2003-2010 Net Insight AB, All Rights Reserved

Adding a Hierarchical Map using Map Group

With a symbol in the map that represents a map-group, you can easily represent a set of symbols with a group symbol. Expanding the group displays the group contents. Because the group is actually a part of the map, the map properties are the same for the group as for the rest of the map. The group symbol color has the color representing the highest severity of any object in the group. Links between symbols inside and outside the map are rendered to and from the group symbol when the group symbol is displayed. You cannot create groups within groups.

To group symbols in a map:

1. Open the map.
2. Select the symbols and links that shall be part of the group.
3. Select the **Group Selected Symbols** tool in the map toolbar. The group is created.

See Working with Map Symbols in the Web NMS User's Guide for details on how to manage groups.

Adding a Symbol to an existing Map Group

You can add a symbol to an already existing map group, or move a symbol from one group to another. To move a symbol:

4. Open the map.
5. Open the Symbol Properties form for the symbol you want to move (**Nimbra Node | Symbol Properties**).
6. Enter the name of the group in the property filed for **Group Name** in the **Relationship Details** section. If the group does not already exist, a new group will be created with that name.
7. Check **Save changes on server** to make the change permanent.
8. Click **Modify**.

Adding a Hierarchical Map using Map Symbol

By having a symbol in the map representing another map, e.g. a sub-map, you can open the sub-map from the parent map. The opened map is displayed in a separate window. Because the sub-map is created separately the parent map, it can be referenced from other maps as well as from the Tree. The map properties for the sub-map are used by sub-map, and are not affected by the parent map's properties. This means that your sub-map can have its own background image, topology, renderers, criteria properties etc. However, you will have to create all the links to/from the sub-map symbols in the parent map manually. The color of the symbol is not affected by the severity of the objects in the map. Using map symbols, you can create hierarchies with any depth.

To create a hierarchical map:

1. Create the sub-maps that shall be included in the parent map. You can use any type of map. See Adding a DTM Map on how to add a DTM Map.
2. Create the parent map. This could be any map that you would like to have as parent map, e.g. an empty map. See Adding a DTM Map on how to create an empty map. Note that you can populate the map with nodes and links as with any other map.
3. Open the parent map.
4. You can cut-and-paste symbols from other maps into the parent map.

© 2003-2010 Net Insight AB, All Rights Reserved

5. To link a symbol in the parent net to another map: In the parent map, create a map symbol that shall represent the sub-map, using menu **Edit | Add Symbol** (or use the toolbar button **Add Symbol**). This must be done for each sub-map that shall be represented by a symbol.
6. For the map symbol, set the properties as:
   Name must be the same as the name of the map that the symbol represents, but without the map name suffix (`.netmap`). The name is what links the symbol to the sub-map. This property cannot be change later.
   ObjName must be `null`, because the symbol represents a map, not a managed object.
   MenuName should be `dtmnet`, but could include other menus as well. This is the pop-up menu for the symbol. To add multiple menus, list them separated by commas.
   IconName could be `dtmnet.png`.
   ObjType could be any of `sub-symbol`, `site`, `background`, `node`, `network` or `gateway`.
   Save changes on server must be set to save the symbol on the server.
7. Click on **Modify** to create the symbol. Opening the symbol will now open the corresponding sub-map.
8. You may have to refresh the map to view the changes that were saved on the server.

To create links to and from the symbols in the parent map:

1. Open the parent map.
2. In the parent map, create a link symbol, using menu **Edit | Add Link** (or the toolbar button **Add Link**).
3. For the link symbol, set the properties:
   name is any name for the link. This must be unique.
   source and **dest** are the names of the originating and terminating symbols.
   menuName should be `dtmlinkmenu.`
   objName must be the name of the link managed object that the symbol shall represent.
   Save changes on server must be set to save the symbol on the server.
4. Click **Modify** to create the symbol.
5. You may have to refresh the map to view the changes that were saved on the server.

Tip! *You can cut-and-paste two nodes with their links from another map, and then modify the* ***source*** *and* ***dest*** *properties for the links.*

See Also

in the Web NMS User's Guide

Exporting and Importing Maps

You can export a map layout, and import a previously exported map layout.

When exporting a map, the map with its properties, and all it symbols properties are exported to an XML file. Note that only data about the map and its symbol

© 2003-2010 Net Insight AB, All Rights Reserved

are saved. E.g. the background image or symbol image are not included, only the references to the images.

To export a map, open the map and choose **Custom View | Export Map…**. This opens a file choose. Enter a file name and click **Save**.

To import a map, open a map and choose **Custom View | Import Map…**. This opens a file chooser. Enter the file with the saved map and click **Open**. You must now select one of:

- **Add a new map with name:** ***map name***. This creates a new map with same properties and symbols as the map saved in the import file. You must provide a unique map name.
- **Add and update symbols in existing map**: ***map name***. This will update the map that was selected when starting the function. Any symbol is the map that also exists in the file will be updated according to file. Any symbol not present in the map, but that exists in the file will be added. Any symbol in the map that does not exist in the file will remain unchanged. The properties of the map itself will also remain unchanged, retaining its background, criteria etc.

Click button **OK** to import the map

Working with Network Database

Network Database

The Network Database maintains the properties of all Managed objects of a network. These managed objects and their object properties are listed in the Network Database panel. The Network Database panel can be selected from the Tree, which is on the left side of Nimbra Vision panel. You can navigate through the displayed Network Database panel by accessing the navigating buttons, thereby sorting the listed Managed Objects. The property details of these Managed Objects can be accessed through the menu specific for the selected object. The Network Database panel operations are triggered on accessing the menus that are specific for the panel.

By default, sets of Custom Views are defined for the Network Database. You can easily define other Custom Views, or modify the exiting ones. The default Custom Views displays the properties that you are most likely interested in for the type of Custom View. Please refer to the managed object properties description for each class of managed object to get a description on its properties.

The table below describes the default Custom Views for the Database.

Name	Description
Networks	IP networks. These are also displayed in the **ipnet** map.
Nodes	All nodes, i.e. the managed objects with the property **isNode** set to *true*.
Interfaces	All interfaces, i.e. the managed objects with the property **isInterface** set to *true*.

© 2003-2010 Net Insight AB, All Rights Reserved

Name	Description
DTM Nodes	All nodes that are Nimbra nodes, i.e. managed objects with the property **classname** set to *DtmNode*. These are also displayed in the DTM Network map. See Nimbra Node Properties for a description of the properties.
DTM Interfaces	All DTM interfaces, i.e. managed objects with the property **classname** set to *DtmInterface*. See DTM Interface Properties for a description of the properties.
DTM Links	All DTM links i.e. managed objects with **classname** set to *DtmLink*. These are also displayed in the **DTM Network** map. See DTM Link Properties for a description of the properties.
Services	All services, i.e. managed objects with the property **classname** set to *DtmService*.

See Also

Traversing the Network Database in the Web NMS User Guide.

Working with Custom Views in the Web NSM User Guide.

Writing Filter Criteria

In many places, filter criteria are used for filtering data to a sub-set of data.

Generally, filtering criteria consists of one or many object properties with filter strings value.

- The string value is absolutely matched, meaning that a word or a phrase is matched exactly as written only, but with the addition that case is ignored and with use of wildcards, see table below.
- If criteria are defined for multiple properties, then the criteria for all the properties must match, i.e. **and** is operated on all the criteria.

Wildcard character	Description
	Empty string means that the property is not included in the match criteria, which means that everything matches this.
*	Asterisk character matches zero or more characters. Can be used in the beginning, within, and at the end of an expression.
!	Exclamation characters negates the following expression, i.e. is a **not** operator. Can only be used as first character of an expression.
,	Comma is used to separate multiple expressions for the same property where any of the expressions must match, i.e. an **or** operator.
&&	Two ampersands are used to separate multiple expressions for the same property where all of the expressions must match, i.e. an **and** operator. The **and** operator has precedence over the **or** operator.
<between> *a* and *b*	Match only values between *a* and *b*, including the limits.

© 2003-2010 Net Insight AB, All Rights Reserved

Example

The following examples show different criteria and examples of what would be matched. The third column lists all matches for the criterion assuming that the following exists: `NimbraOne`, `Nimbra340`, `Nimbra680` and `SnmpNode`.

Wildcard character	Description	Match example
	Ignored in match criteria, which means that everything is matched.	NimbraOne Nimbra340 Nimbra680 SnmpNode
`NimbraOne`	Matches only NimbraOne, and nothing else.	NimbraOne
`NIMBRAONE`	Matches only NimbraOne, and nothing else.	NimbraOne
`Nimbra`	Matches only Nimbra, and nothing else, e.g. nothing would match in this example.	
`Nimbra*`	Matches anything beginning with Nimbra, e.g. NimbraOne, but also Nimbra340, Nimbra680 etc.	NimbraOne Nimbra340 Nimbra680
`Nimbra*0`	Matches anything beginning with Nimbra and ending with 0, e.g. Nimbra340, Nimbra680, but not NimbraOne.	Nimbra340 Nimbra680
`NimbraOne,Nimbra340`	Matches any of NimbraOne and Nimbra340, but nothing else.	NimbraOne Nimbra340
`!NimbraOne`	Matches anything except NimbraOne, e.g. Nimbra340, Nimbra680, SnmpNode etc.	Nimbra340 Nimbra680 SnmpNode
`Nimbra*&&!NimbraOne`	Matches anything beginning with Nimbra except NimbraOne, e.g. Nimra340, Nimbra680 but not SnmpNode.	Nimbra340 Nimbra680
`<between> 10 and 20`	Matches only with numeric values from 10 to 20 (inclusive). This is only applicable for numerical properties.	

See Also

Appendix F: Custom View Properties, Tips and Tricks in the AdventNet Web NMS User Guide for help on writing criteria.

© 2003-2010 Net Insight AB, All Rights Reserved

Device Based Operations

Operations on DTM Nodes

Introduction to Operations on Nimbra Nodes

You can perform different operations on Nimbra nodes, the managed objects representing the nodes, or the symbols in the map that represents the managed objects.

To perform an operation, select the Nimbra node in the map, or in the network database. This will enable the object menu **Nimbra Node**. You can also use the right mouse button on the object to open the menu as a pop-up menu.

The available operations depend on your permissions, as defined by the administrator.

Operations

- Element Manager. Opens the web browser and loads the node's element manager.
- System Information... Opens a dialog for some basic information about the system.
- DTM Interfaces... Present a list of all DTM interfaces.
- Node Configurations... Manage configuration files on the node.
- Save Configuration... Save the current configuration on a node on a local file. You can select multiple nodes to perform the operation on multiple nodes.
- Add Service... Opens wizard to add a new ITS or ETS service that originates in the node.
- Edit Source Routes... Opens an editor for adding or modifying source routes defined in the node.
- Edit VLAN Configurations... Open an editor for adding or modifying VLANs, and their associations to Ethernet interfaces and ETS Services.
- **List Channels...** Present a list of all channels originating, transiting or terminating on the node.
- **Trace Channel...** Traces a channel originating in the node.
- Trace Sync. . . . Traces the synchronization and time transfer in all or part of the network.
- Telnet to Device. Opens a telnet session to the node for Command Line Interface (CLI) access.
- Ping. Send ICMP ECHO_REQUEST packets to the node using the management network.
- Trace Route. Print the route packets take to a node on the management network.
- Symbol Properties. Opens a dialog to modify the properties for the map symbol.
- Managed Object Properties. Opens a dialog to modify the properties of the managed object.

© 2003-2010 Net Insight AB, All Rights Reserved

- Set Tag. . . . Opens a dialog to modify the tag property on a node and optionally its child objects.
- Update Status. Immediately makes a status poll of the node.
- Refresh. Reads all the information from the node and updates the database and maps.
- **Delete from Database**. Deletes the node from the database, including its interfaces and links.
- Manage. Sets the managed status to managed.
- Unmanage. Sets the managed status to unmanged. When a node unmanaged, it is ignored by Nimbra Vision.

Element Manager

The **Element Manager** operation opens the web browser and connects to the node. This will display the node's web interface, i.e. the node's element manager. The HTTP session is setup between your web browser and the node. The Nimbra Vision server is not involved.

See Also

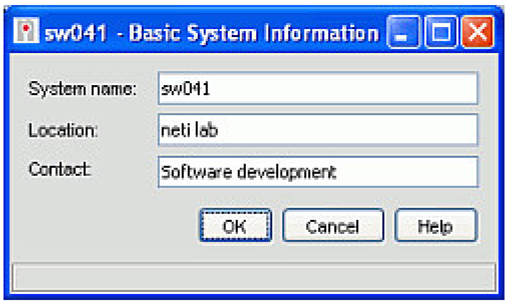
Telnet to device

System Information

The System Information dialog shows some basic system information about the node.

The system information is part of the standard information that is mandatory for all SNMP devices. The information is included in the SNMPv2-MIB, defined in RFC3418.

To open the System Information dialog, select a node in the DTM Network map or in the Network Database, and select the object menu **Nimbra Node | System Information…**.


sw041 - Basic System Information

System name: sw041

Location: neti lab

Contact: Software development

System information.

Field descriptions

- **System name**. An administratively assigned name for the node. By convention, this is the node's fully qualified domain name, as known in the IP network.
- **Location**. Textual description of the physical location of the node. This can be used to describe where a node is located.
- **Contact**. The textual identification of the contact person for this node, together with information on how to contact this person.

Click **OK** to apply the changes and close the dialog.

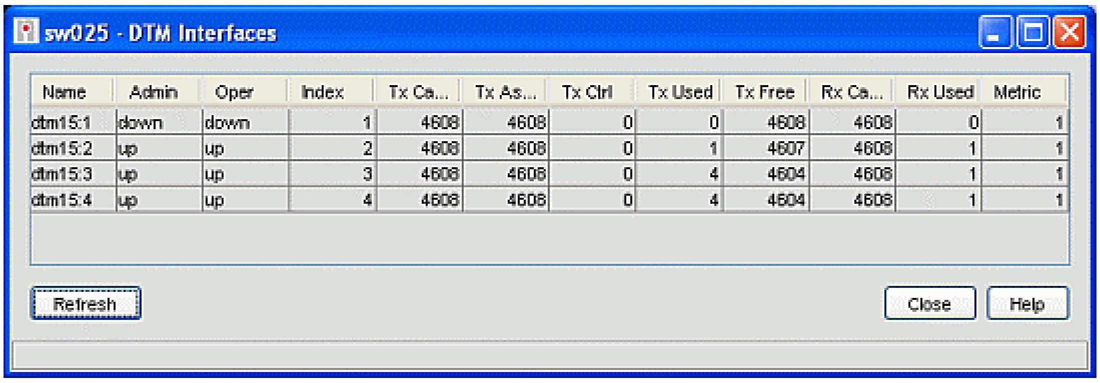
Click **Close** to close the dialog without applying any changes.

© 2003-2010 Net Insight AB, All Rights Reserved

List of DTM Interfaces

The DTM Interfaces dialog shows a summary of all the DTM interfaces on a node as a table. The information is about the same as for a single interface that is displayed in the Configure DTM Interface dialog. It is retrieved directly from the node.

To open the DTM Interfaces dialog, select a node in the DTM Network map or in the Network Database, and select the object menu **Nimbra Node | DTM Interfaces…**.



Name	Admin	Oper	Index	Tx Ca...	Tx As...	Tx Ctrl	Tx Used	Tx Free	Rx Ca...	Rx Used	Metric
dtm15:1	down	down	1	4608	4608	0	0	4608	4608	0	1
dtm15:2	up	up	2	4608	4608	0	1	4607	4608	1	1
dtm15:3	up	up	3	4608	4608	0	4	4604	4608	1	1
dtm15:4	up	up	4	4608	4608	0	4	4604	4608	1	1

List of DTM interfaces

- **Name**. The local name of the interface, as named by the node. This name is unique within the node. Typically, the name consists of the string *dtm* followed by the board position number and port number at the board.
- **Admin**. The administrative state of the interface.
- **Oper**. The operational state of the interface.
- **Index**. The index of the interface in the node's interface table.
- **Tx Capacity**. The capacity of the interface in number of slots. The capacity is given by the underlying trunk interface.
- **Tx Assigned**.. The assigned capacity of the interface. Normally, the complete interface capacity is assigned for usage, but it is possible assign a smaller capacity to an interface.
- **Tx Ctrl**. The number of slots used for control data, except slot 0.
- **Tx Used**. The number of slots used for transmit traffic.
- **Tx Free**. The number of free slots for transmit traffic. This is the number of assigned slots that are not used.
- **Rx Capacity**. The receive capacity of the interface in number of slots. The capacity is given by the underlying trunk interface.
- **Rx Used**. The number of slots used for receive traffic.
- **Metric**. The metric (cost) on this interface, used for Dynamic Routing Protocol (DRP).

Click **Refresh** to update the dialog. The information is retrieved directly from the node.

Click **Close** to close the dialog.

See Also

Configure DTM Interface

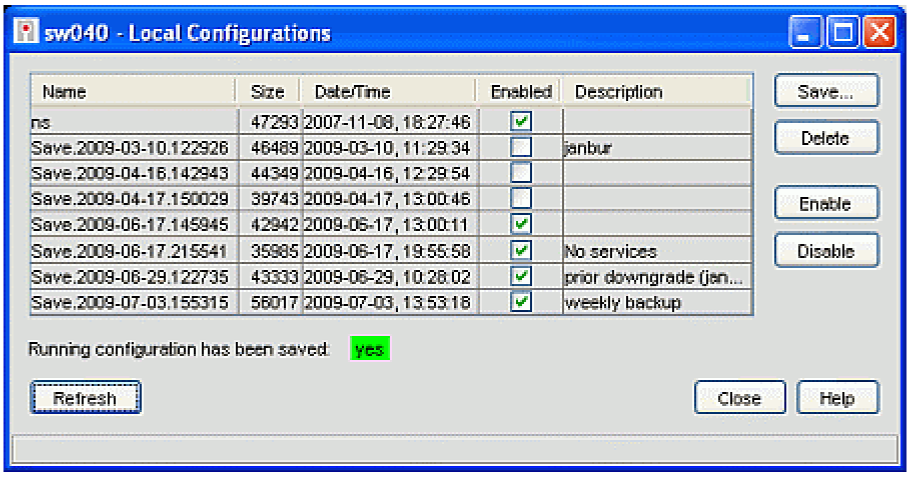
© 2003-2010 Net Insight AB, All Rights Reserved

Managing Local Configuration Files

The Local Configurations window allows you to manage local configuration files in the node. A configuration file contains the complete configuration of the node. This includes the configuration of any services. When the node is booting, it reads the newest configuration file that is *enabled*, and configures itself accordingly.

A node may have multiple configuration files. The maximum number of saved configurations on a node depends on the capabilities of the node and the size of the configuration files.

To open the window, select a node in the DTM Network map or in the Network Database, and select the object menu **Nimbra Node | Node configurations…**.



Name	Size	Date/Time	Enabled	Description
ns	47293	2007-11-08, 18:27:46	✓	
Save.2009-03-10.122926	46489	2009-03-10, 11:29:34	☐	janbur
Save.2009-04-16.142943	44349	2009-04-16, 12:29:54	☐	
Save.2009-04-17.150029	39743	2009-04-17, 13:00:46	☐	
Save.2009-06-17.145945	42942	2009-06-17, 13:00:11	✓	
Save.2009-06-17.215541	35985	2009-06-17, 19:55:58	✓	No services
Save.2009-06-29.122735	43333	2009-06-29, 10:28:02	✓	prior downgrade (jan...
Save.2009-07-03.155315	56017	2009-07-03, 13:53:18	✓	weekly backup

Running configuration has been saved: yes

Manage the configurations on a node.

The window shows a table with all the configuration files and their attributes:

- **Name**. A name of the file. The name must be unique among the node's configuration files.
- **Size**. The configuration file's size in bytes.
- **Date/Time**. A time stamp when the configuration file was created. The node creates the time stamp when it saves its configuration. The node uses its local calendar clock.
- **Enabled**. The status of the configuration file. A configuration file may be *enabled* or *disabled*. When the node boots, it is reading the newest configuration file with status *enabled*.
- **Description**. A textual description of the configuration.

Running configuration has been saved. This indicates that the node's running configuration has been saved, and that it has not been changed since it was last saved. The node detects that the running configuration is changed when any configuration data has been modified in the node.

Click **Save…** to save the node's current configuration into a new configurations file. The file will have the status *enabled*. This means that the node will use this configuration file next time it is booting.

Click **Delete** to delete the configuration file that is currently selected in the table.

Click **Enable** to enable the configuration file that is currently selected.

Click **Disable** to disable the configuration file that is currently selected. The node ignores a *disabled* configuration file at boot.

© 2003-2010 Net Insight AB, All Rights Reserved

Click **Refresh** to update the window. The information is retrieved directly from the node. Use **Refresh** if someone else has modified the table since the window was opened.

Click **Close** to close the window.

See Also

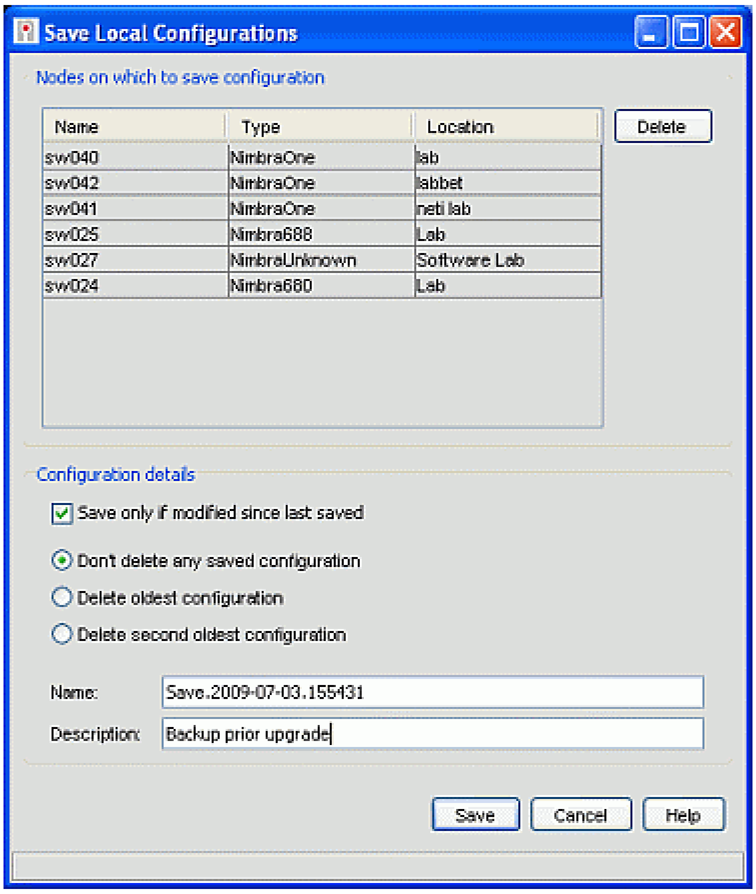
Save Local Configuration on Multiple Nodes

Save DTM Node Registry - a policy to save configurations according to a scheduler

Save Local Configuration on Multiple Nodes

The dialog Save configurations on multiple nodes allows you to save the current configuration of multiple nodes to local files. For each node, its current configuration will be saved in a local file stored on the node. Note that there is a limit of the number of configurations that can be saved on a node. The limit depends on the capabilities of the node, and the size of the saved configurations.

To open the window, in the DTM Network map or the Network Database, select all the nodes for which the configuration shall be saved, and select the object menu **Nimbra Node | Save configuration…**.


Save Local Configurations
Nodes on which to save configuration
Delete
Configuration details
Save
Cancel
Help

Name	Type	Location
sw040	NimbraOne	lab
sw042	NimbraOne	labbet
sw041	NimbraOne	neti lab
sw025	Nimbra688	Lab
sw027	NimbraUnknown	Software Lab
sw024	Nimbra680	Lab

✓ Save only if modified since last saved

[✓ Don't delete any saved configuration]
Delete oldest configuration
Delete second oldest configuration

Name: Save.2009-07-03.155431
Description: Backup prior upgrade

Save the current (running) configuration on one or many nodes.

The list contains all nodes for which the configuration shall be saved. The list contains all the nodes that were selected in the DTM Network map or the Network Database when the window was opened.

Click **Delete** to remove a selected entry from the table. This could be used if you, when you opened the window, selected a node for which you don't want to save the configuration.

© 2003-2010 Net Insight AB, All Rights Reserved

- **Save only if modified since last saved**. Selecting this option, the function checks if the current configuration in the node has changed since it was last saved. If it has changed, the configuration will be saved. If it has not changed, the configuration is not saved.
- **Don't delete any saved configuration**. The existing configurations on the node are untouched.
- **Delete the oldest saved configuration**. If a configuration shall be saved, the oldest saved configuration on the node is removed prior the current configuration is saved. Removing an old configuration makes room for new configurations. A configuration is removed only if a new configuration is saved.
- **Delete the second oldest saved configuration**. As above, but the second oldest configuration is removed instead of the oldest. This can be used to keep one configuration (the oldest) as a fall-back configuration that is always available.
- **Name**. The name of the configuration file. The name must be unique within the node. The field will contain a suggested name that is likely to be unique as it is based on the current time. The same name will be used on all nodes.
- **Description**. A textual description of the configuration that will be stored with the configuration.

Click **Save** to save the current configuration on all the nodes in the list.

Click **Cancel** to close the window without saving any configurations.

Note! *There are a maximum number of configurations that can be saved on the node. The amount depends on the type of node, and sometimes the size of the configuration files. Removing an old configuration prior saving the current configuration will ensure that the configuration is saved.*

See Also

Managing Local Configuration Files

Save DTM Node Registry - a policy to save configurations according to a scheduler

List Channels

List Channels is a function that retrieves information from a node about all channels in the node. This includes originating, transiting and terminating channels. The function lists all the channels a new window. The function always retrieves data about all the channels in the node, but it is possible to filter what channels to display.

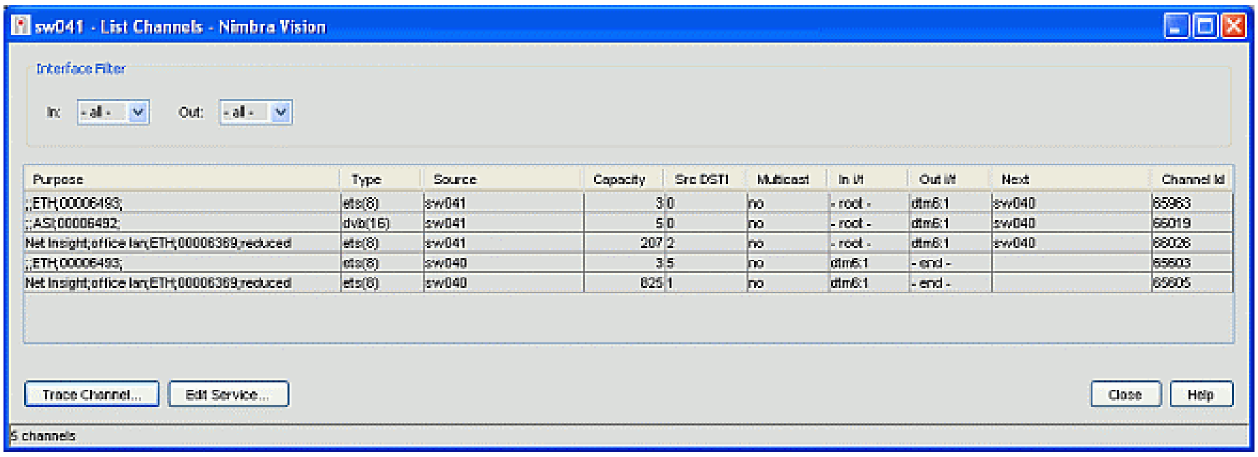
- To list all channels in a node, select the node and choose menu **Nimbra Node | List Channels…**.
- To list all channels on a link, select the link and choose menu **Link | List Channels…**. The filter is set so that only channels leaving the node on the specified link are displayed.
- To list all channels on a bi-directional link, select the link and choose menu **Link | List Channels (Bi-directional)…**. The filter is set so that only channels leaving or entering the node on the selected link or its return link are displayed.
- To list all channels on an interface, select the interface and choose menu **Interface | List Channels…**. The filter is set so that only channels leaving the node on the specified interface are displayed. Note

© 2003-2010 Net Insight AB, All Rights Reserved

that listing channels on an interface is the same as listing channels on a link that is originating in the interface.

- To view all channels on a bi-directional interface, select the interface and choose menu **Interface | List Channels (Bi-directional)…**. The filter is set so that only channels leaving or entering the node on the selected interface are displayed. Note that listing channels on an interface is the same as listing channels on a link that is originating in the interface.

You can filter displayed channels using the **Interface Filter**. Only channels matching the selected In and Out interfaces are displayed. Choose "`- all -`" to match any interface, choose "`- root -`" to match channel originating on the node, and choose "`- end -`" to match channels terminating on the node, If the same interface is selected for In and Out, then channels either entering or exiting the interface are displayed; this is how a bi-directional link is filtered.



Purpose	Type	Source	Capacity	Src DSTI	Multicast	In i/f	Out i/f	Next	Channel Id
;;ETH;00006493;	ets(8)	sw041	3	0	no	- root -	dtm6:1	sw040	65963
;;ASI;00006492;	dvb(16)	sw041	5	0	no	- root -	dtm6:1	sw040	66019
Net Insight;office lan;ETH;00006369;reduced	ets(8)	sw041	207	2	no	- root -	dtm6:1	sw040	66026
;;ETH;00006493;	ets(8)	sw040	3	5	no	dtm6:1	- end -		65603
Net Insight;office lan;ETH;00006369;reduced	ets(8)	sw040	825	1	no	dtm6:1	- end -		65605

5 channels

The list channels dialog.

You can trace a specific channel through the network by selecting a channel in the table and click button **Trace Channel…**. See Trace Channel.

You can open the Service Editor for the service that is using a specific channel by selecting the channel in the table and click button **Edit Service…**. See Editing a Service.

The following data are displayed for a channel.

- **Purpose** is the purpose as configured in the service (requires Service Provisioning).
- **Type** is the type of service (DST) for the connection. See table below.
- **Source** is the name of the node where the channel originates.
- **Capacity** is the number of 512 kbps slots.
- **Src DSTI** is the DSTI for the TTP where the channel originates.
- **Multicast** is "yes" in case of a multicast channel, otherwise "no". Note that a multicast channel does not necessarily split in multiple branched on the queried node.
- **In i/f** is the name of the DTM interface where the channel is entering the node. If the channel originates at the node, this is indicated as "-root -".
- **Out i/f** is the name of the DTM interface where the channel exits the node. If this is the last node for the channel or for the branch on a multicast channel, this is indicated as "- end -". A multicast channel

© 2003-2010 Net Insight AB, All Rights Reserved

that is split in the node exits the node through multiple interfaces. These are all represented as separate rows in the table. If a multicast channel is terminated in the node and also exits the node, then the local termination will not be displayed.

- **Next** is the name of the next node for the channel of branch, i.e. the node where the out interface is connected.
- **Channel Id** is an identity number for the channel. This number is unique in the node where the channel originates. The source together with the channel id is a unique identifier of the channel in the complete network.

Type of Service (DST)	Description
0	Control channel
1	DLE (DTM LAN Emulation) control and data
2	DLE control only
3	DLE data only
4	PDH transport
5	DLE Inter server control and data
6	DLE Inter server control
7	DLE Inter server data
8	ETS (Ethernet Transport Service). The ETSI standardized name for this service is DLT (DTM LAN Transport)
9	Trace route channel
10	IPOD channel
11	loadgen
12	DTM ping
13	E1
14	DS1 (T1)
15	BT 601 (formerly called SDI 270)
16	DVB (formerly called ASI)
17	Point-to-multipoint test application
18	SDH
19	TTCP (Time Transfer Channel Protocol)
20	AES/EBU

See Also

Trace Channel

Editing a Service

Trace Channel

Trace Channel is a function that traces a channel through the network, from the node where the channel originates, and to all the terminating nodes (multiple if multicast).

© 2003-2010 Net Insight AB, All Rights Reserved

Note that a 1+1 protected service is implemented using two independent channels through the network. These two channels have different channel identities. You can run multiple Trace Channels (e.g. one for each channel for a 1+1 protected service).

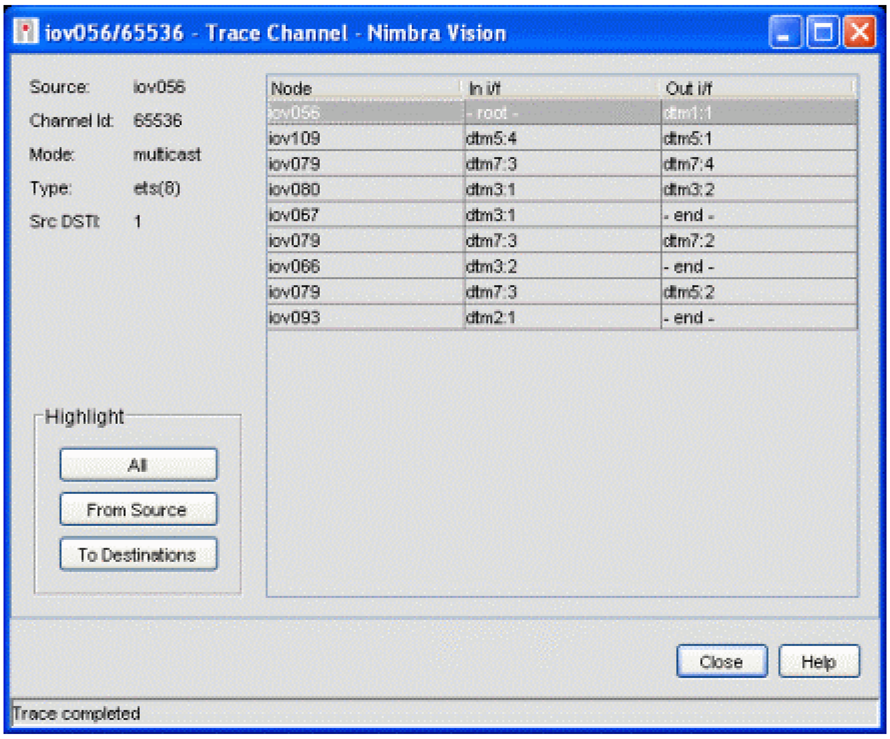
Starting to Trace a Channel

You can start the trace of a channel from a selected channel in a channel list result, see List Channels.

If you already know the identity of the channel, i.e. the name of the node where the channel originates, and the channel id number, then you can start a channel trace based on this information. The channel id of the channel could be retrieved from the Element Manager. To start a channel trace, do:

- Select a node in a map or the network database, and select menu **Nimbra Node | Trace Channel….** A dialog is opened
- Enter the name of the node where the channel originates (this defaults to the name of the node that was selected when the menu was selected). Select the ellipsis button (…) to open a node search dialog.
- Enter the id of the channel.
- Click button **Ok**.

If you have Service Provisioning, the you can also trace all channels belonging to a service by selecting the service in the network database and select menu **Service | Trace Channel…**.



Source: iov056
Channel Id: 65536
Mode: multicast
Type: ets(8)
Src DSTI: 1

Node	In i/f	Out i/f
iov056	- root -	dtm1:1
iov109	dtm5:4	dtm5:1
iov079	dtm7:3	dtm7:4
iov080	dtm3:1	dtm3:2
iov067	dtm3:1	- end -
iov079	dtm7:3	dtm7:2
iov066	dtm3:2	- end -
iov079	dtm7:3	dtm5:2
iov093	dtm2:1	- end -

Trace completed

The trace channel dialog.

How Channel Trace Works

The function trace channel traces a channel through the network. It starts by querying the node where the channel originates for the channel's next-hop node (or, potentially nodes in case that a multicast channel is forked). It continues by querying the next-hop node about where the channel continues, and so on, until the channel is completely traced through the network.

© 2003-2010 Net Insight AB, All Rights Reserved

The result is displayed as an ordered list of next-hop nodes, starting with the node where the channel originates. The result can also be displayed on all the maps as a highlight.

For each next-hop, the following data is displayed:

- **Node** is the name of the node where the channel exists.
- **In i/f** is the name of the DTM interface where the channel branch enters the node. The channel originates at the first node in the list, which is indicated with "- root -".
- **Out i/f** is the name of the DTM interface where the channel branch exits the node. If the channel branch terminates in the node, this is indicated with "- end -".

When a multicast channel is traced, then a main branch (trunk) of the channel is first listed, where the last entry of the channel trunk is indicated with "- end -" as out interface. Each branch of the channel is then appended in sequence, where each branch's first entry is the name of the node where the branch is forked, and its last entry is indicated with "- end -" as out interface. Generally, the trunk of a channel is the path where the out interface with the lexographically lowest interface name is used.

In addition to the path of the traced channel, the following data is also displayed about the channel:

- **Source** is the name of the node where the channel originates.
- **Channel Id** is an identity number for the channel. This number is unique in the node where the channel originates. The source together with the channel id is a unique identifier of the channel in the complete network.
- **Type** is the type of service (DST) for the connection. Also see List Channels.
- **Src DSTI** is the DSTI for the TTP where the channel originates.
- **Mode** is "unicast" in case of a unicast channel, or "multicast" in case of a multicast channel.
- **Capaciy**. The capacity of the channel in 512 kbps slots.
- **Purpose**. The purpose of the service to which this channel belongs.

Display Result in Map

The channel path can be visually displayed in all maps as highlighted symbols. To highlight the channel path in the maps, select a button in **Highlight**. This highlights all nodes and links in the map according to result of the channel trace and the selected button. The buttons appends to any already highlighted nodes and links. This means that you can show the result of multiple channel trace simultaneously (e.g. the two channels for an 1+1 protected service).

To clear all the highlights in the maps, select the menu **Edit | Clear Highlight** or the **Clear Highlight** tool bar button, which are available when a map is selected.

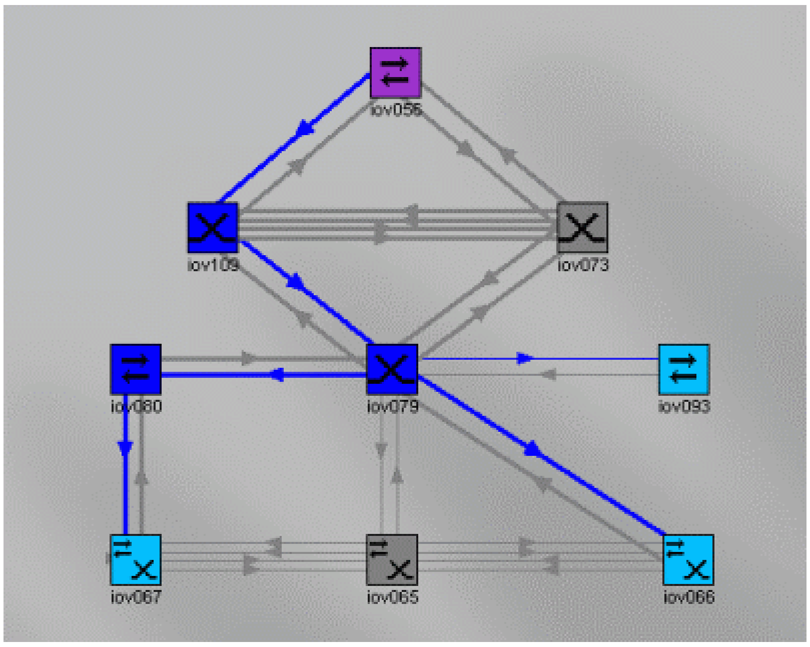
You can highlight the complete channel, or a part of the channel:

Click **All** to highlight the complete channel.

Click **From Source** to highlight the path from the originating node (source) to the node selected in the list.

Click **To Destinations** to highlight the sub-tree from the node selected in the list to all destinations below the node.

© 2003-2010 Net Insight AB, All Rights Reserved



Channel highlighted in map

When highlighting a channel, the following colors are used:

Purpose	Color Name	Color
Channel originates (root)	Dark Orchid	🟪
Channel is transits	Blue	🟦
Channel terminates	Deep Sky Blue	🟦

See Also

List Channels

Telnet to Device

The **Telnet** operation opens a telnet session to the node. From the telnet session, you can run the Command Line Interface (CLI).

The telnet session is opened between the Nimbra Vision server and the node, but is displayed in your client. You do not need to have telnet access from your client computer to the node to be able to run this operation: you can run the telnet operation even if the DTM nodes are behind a firewall that does not allow telnet services.

See Also

Element Manager

Ping

The **Ping** operation sends ICMP ECHO_REQUEST packets to the node, and displays the result in a window. **Ping** can be used to check connectivity to a remote node on the IP network.

***Note!* Ping** *is using the IP network, not the DTM network. It does not operate on the DTM network level, and is not directly affected by the status of the DTM network. However, if DLE in-band management network is used, the DTM network and the status of the DLE service affect the IP network.*

© 2003-2010 Net Insight AB, All Rights Reserved

Trace Route

The **Trace Route** operation print the route packets take to a node on the IP management network. The information is displayed in a window. **Trace Route** can be used for trouble shooting problems in the management network.

> ***Note!* *Trace Route*** *is using the IP network, not the DTM network. It does not operate on the DTM network level, and is not directly affected by the status of the DTM network. However, if DLE in-band management network is used, the DTM network and the status of the DLE service affect the IP network.*

Tagging Managed Objects

The `tag` property of a managed object can be used to administratively or logically group managed objects. The tag is simply a property on the managed object that can have an arbitrary string. You can assign different tags to different managed objects an then use this tag in filters, custom views, custom maps and other criteria based functions to filter out a set of managed objects related to any user defined domain represented by the tag.

One way to use the tag is to let it represent different geographical regions. Use the tag as a criterion in maps and Network Database custom views to easily populate them with managed objects only related to the geographical region.

Another usage of the tag is to let it represent different organizations.

If you want to a managed object to be part of multiple logical groups (e.g. a node is included in more than one region, perhaps a geographical region and an administrative region), then you could add multiple words (or sub-strings) in tag, where each represents its logical group. You would then need to surround the match criteria with wildcards in custom views etc.

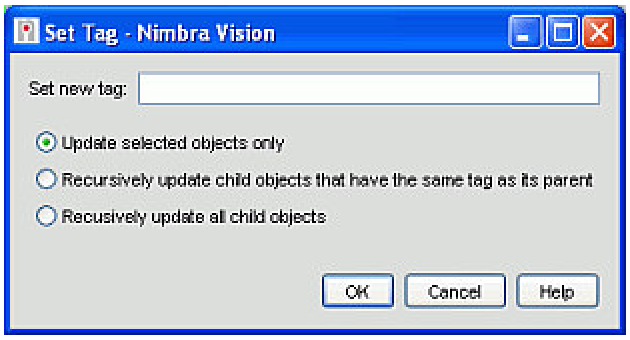
> ***Note!*** *Scopes does not support wild cards, so you cannot use multiple words in tags to represent multiple logical groups if fine-grained authorization is to be applied on operations on the object. See Security Management.*

Nimbra Vision maintains a logical parent-child relationship between some classes of managed objects when it comes to tags:

- A `DtmLink` is a child of the terminating `DtmInterface`. When a new link is discovered, its tag is automatically set to the same value as its parent `DtmLinterface`.
- A `DtmInterface` is a child of its node. When a new interface is discovered, its tag is automatically set to the same value as its parent `DtmNode`.
- A `DtmService` is a child of its originating node. When a new service is discovered (i.e. added), its tag is automatically set to the same value as its parent `DtmNode`.
- Any other managed object does not have a parent relationship when it comes to tags. When such a managed object is newly discovered, its tag is an empty string.
- An event gets its tag set to the same value as the managed object to which the event belongs.
- An alarm gets its tag set to the save value as of the event that last updated the alarm.

To set or change a tag for a selection of managed objects, select the objects in a map or in the Network Database and choose the object menu ***Object* | Set Tag…**. This opens a dialog where you can set the tag for the selected objects and its child objects.

© 2003-2010 Net Insight AB, All Rights Reserved


Set Tag - Nimbra Vision

Set new tag:

(●) Update selected objects only

( ) Recursively update child objects that have the same tag as its parent

( ) Recusively update all child objects

Dialog for setting the tag. The tag is set on the selected objects, and optionally on child objects.

Set new tag. A text string. This is the tag.

Radio buttons to control how to updated the tag.

- **Update selected objects only**. The tag will be set on all the selected managed objects, but no other objects. No events or alarm are affected.
- **Recursively update child objects that have the same tag as parent**. The tag will be set on all selected managed objects, and on child objects if the child objects current tag is the same as its parent's current tag. This option is useful if you wan to keep an existing tag structure on related managed objects. E.g. if you wan to replace a tag string with a new string, then this option is useful. No events or alarm are affected.
- **Recursively updated all child objects**. The tag will be set on all selected objects, and on all its child objects. No events or alarm are affected.

It is also possible to set the tag by setting the property on the managed object using the Managed Objects Properties editor. This will always set the tag on that managed object only.

Update Status

The Update Status operation makes an immediate status poll of the node.

Refresh

The **Refresh** operation re-reads all the information from the node and updates the database and maps accordingly. The operation is a re-discovery of the node. When a node is refreshed, all the properties for the managed object are updated, and all alarms are cleared and reset to the status of the alarms in the node. All the child objects, i.e. all the interfaces and their links are also updated.

The links associated with the object has its administrative name and its bundle name preserved. These two properties are not modified when the managed object is refreshed.

A refresh operation should normally never be required if the node sends SNMP notifications. However, if a node does not send SNMP notifications, there is no mechanism to determine if the status is updated on the node. In this case, the status must be updated using the refresh operation.

If the Discovery Engine has been setup to discover the node as an SNMPv3 device, then the SNMPv3 data is also updated in the **SNMP V3 Security** database.

Manage and Unmanage a Node

Managing a node means that you are making Nimbra Vision to monitor its performance. Unmanaging the node stops Nimbra Vision to monitor the element.

© 2003-2010 Net Insight AB, All Rights Reserved

When a node is unmanaged, Nimbra Vision does not periodically poll the element for its status. Nimbra Vision ignores SNMP notifications from the node that otherwise would result in some kind of status update, including alarm updates.

When you change the status from unmanaged to managed, Nimbra Vision will immediately update the status of the node, which in its turn could result in a refresh operation.

You can perform the **Manage/UnManage** operation on an individual node, or on a complete IP network.

When a node is unmanaged, its status becomes **Unknown**.

Operations on DTM Interfaces

Introduction to Operations on Nimbra Interfaces

You can perform different operations on the DTM interfaces, and the symbol representing the link in the Nimbra map.

To perform an operation, select the DTM interface in the network database. This will enable the object menu **Interface**. You can also use the right mouse button on the object to open the menu as a pop-up menu.

The available operations depend on your permissions, as defined by the administrator.

Operations

- DTM Interface... Opens a dialog to configure the DTM interface.
- List Channels. . . . List channels on the node where the interface is located, and presets the filter to display only channels through the originating interface.
- List Channels (Bi-directional). . . . List channels on the node where the interface is located, and presets the filter to display only channels through the originating or terminating interface.
- Delete from Database. Deletes the interface from the database
- Set Tag. . . . Opens a dialog to modify the tag property on an interface and optionally its related links.
- Managed Object Properties. Opens a dialog to modify the properties of the managed object.

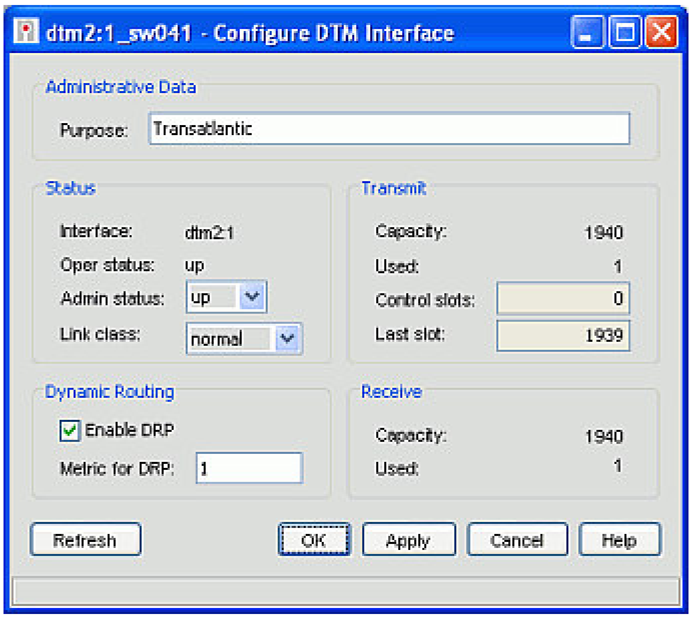
Configure DTM Interface

The DTM Interface configuration dialog allows you to configure a single DTM interface on the node. A DTM interface is responsible for the allocation of link capacity that is originating from the interface.

***Warning!** Changing the configuration of an interface affects the services that use the interface. It could also affect the in-band management network, resulting in that you could potentially loose connectivity with the node.*

To open the dialog, select the DTM interface in the Network Database, and select the object menu **Interface | DTM Interface…**, or select the link in the DTM Network map and select the object menu
Link | Originating DTM Interface… or
Link | Terminating DTM Interface….

© 2003-2010 Net Insight AB, All Rights Reserved


dtm2:1_sw041 - Configure DTM Interface

Administrative Data

Purpose: Transatlantic

Status

Interface: dtm2:1
Oper status: up
Admin status: up
Link class: normal

Transmit

Capacity: 1940
Used: 1
Control slots: 0
Last slot: 1939

Dynamic Routing

[✓] Enable DRP
Metric for DRP: 1

Receive

Capacity: 1940
Used: 1

DTM Interface Configuration dialog.

Field descriptions

The dialog is divided into three areas: *Status*, *Transmit* and *Receive*.

Administrative Data

Purpose. A text describing the purpose of the interface. The text is saved in the network element.

Status

The status area describes general interface information.

- **Interface**. The name of the DTM interface, local to the node. The format of the name is normally the prefix *dtm* followed by the equivalence to the physical board position and the port position on the board, e.g. *dtm6:1* for port #1 on board in position #6.
- **Oper status**. The operational state of the interface. The operational state of the interface could be *up* (the interface is operational), *down* (the interface is not operational), *lowerLayerDown* (the underlying physical interface is not available for traffic) or *absent* (the interface is not present or cannot be detected).
- **Admin status**. The administrative state of the interface. This is the desired state of the interface, which could be *up* or *down*, i.e. the interface can be turned on or off.
- **Link class**. The persistence class for channels over this link, as *normal*, *persistent* or *nailed*. See Channel Persistence.

Dynamic Routing

Settings for the Dynamic Routing Protocol (DRP) on the interface:

- **Enable DRP**. Enable the Dynamic Routing Protocol to propagate its routing data over this interface. This should normally be enabled. It can be disabled when DRP should not be used via specific the interface, e.g. if that interface connects to another operator and you don't want to exchange routing information automatically with that operator. Disabling DRP sets is also equivalent to setting the metric to infinite.

© 2003-2010 Net Insight AB, All Rights Reserved

- **Metric for DRP**. Metric (cost) for using this transmit interface, used by the Dynamic Routing Protocol (DRP).

Transmit

The transmit area contains parameters for the transmit part of the interface.

- **Capacity**. The maximum number of slots that the interface is capable of, including slot 0. This is determined by the underlying trunk interface.
- **Used**. The number of slots that are currently used on the link that originating on the interface. This includes slots for the control channel and the services, including slot 0.
- **Control slots**. The number of slots allocated for a control channel in addition to the slot 0 control channel. The control channel is allocated from the slot range used by the interface. The control channel slots cannot be used for any payload data; it is used for signaling only. Whether this control channel can be used by the nodes or not depends on the version of the node’s system software. This value should normally be set to 0.
- **Last slot**. The last slot of the assigned slot range that is used by the link originating on the interface. Normally, all slots shall be assigned to the interface, but in some cases it may be desired to reduce the capacity. It is recommended to set this value to one less than the transmit capacity to use all the capacity.

Receive

The receive area contains status parameters for the receive part of the interface.

- **Capacity**. The maximum number of slots that the interface is capable of, including slot 0.
- **Used**. The number of slots that are currently used on the link terminating on the interface. This includes slots for the control channel and the services, including slot 0.

Click **Apply** to immediately apply the new data in the parameter fields to the node.

Click **Refresh** to update the data in the dialog. The information is retrieved directly from the node.

Click **Close** to close the dialog without applying any changes.

See Also

Channel Persistence

Operations on DTM Links

Introduction to Operations on Nimbra Links

You can perform different operations on the link between the Nimbra nodes, and on the symbol representing the link in the Nimbra maps.

To perform an operation, select the link in the map, or in the network database. This will enable the object menu **Link**. You can also use the right mouse button on the object to open the menu as a pop-up menu.

The available operations depend on your permissions, as defined by the administrator.

© 2003-2010 Net Insight AB, All Rights Reserved

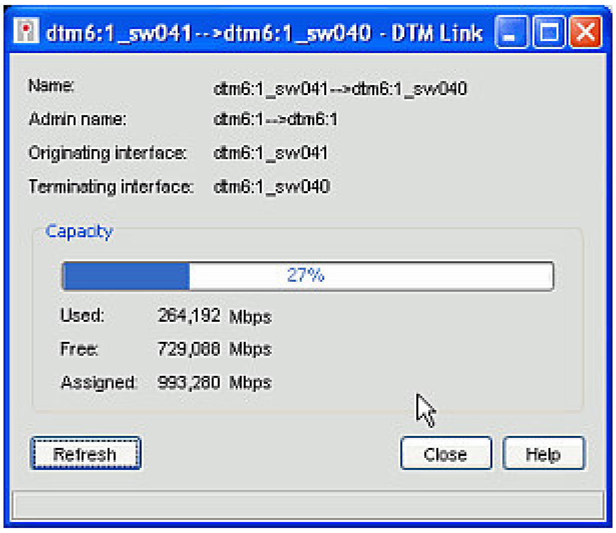
Operations

- DTM Link... Presents information about the link.
- **Originating DTM Interface...** Opens the DTM interface configuration dialog for the link's originating interface.
- Terminating DTM Interface... Opens the DTM interface configuration dialog for the link's terminating interface.
- List Channels. . . . List channels on the node where the link originates, and presets the filter to display only channels on the selected link.
- List Channels (Bi-directional). . . . List channels on the node where the link originates, and presets the filter to display only channels on the selected link or its return link.
- Set Tag. . . . Opens a dialog to modify the tag property on a link.
- Managed Object Properties. Opens a dialog to modify the properties of the managed object.
- Delete from Database. Deletes the link from the map and database.

DTM Link

The DTM Link window shows data about a DTM link. A DTM interface is a logical interface responsible for the slot allocation. Typically, a DTM interface corresponds to one physical trunk interface. When a DTM interface corresponds to exactly one trunk interface, the DTM link represents the physical connection between two adjacent nodes.

To open the DTM Link window, select a link in the DTM Network map or in the Network Database, and select the object menu **Link | DTM Link…**. The data is retrieved directly from the node.


dtm6:1_sw041-->dtm6:1_sw040 - DTM Link

Name: dtm6:1_sw041-->dtm6:1_sw040
Admin name: dtm6:1-->dtm6:1
Originating interface: dtm6:1_sw041
Terminating interface: dtm6:1_sw040

Capacity
27%
Used: 264,192 Mbps
Free: 729,088 Mbps
Assigned: 993,280 Mbps

Display the DTM link data.

- **Name**. The name of the link in Nimbra Vision.
- **Admin name**. This is an administrative name for the link. When the link is created, the system assigns an administrative name to the link. This is a property of the DTM Link managed object, which can be changed by the user.
- **Originating interface**. The name in Nimbra Vision of the originating interface.

© 2003-2010 Net Insight AB, All Rights Reserved

- **Terminating interface**. The name in Nimbra Vision of the terminating interface
- **Usage bar**. The usage bar shows the ratio of used capacity compared to the assigned capacity. This is the load of the link.
- **Used**. The capacity in Mbps that is used on the originating interface.'
- **Free**. The free capacity in Mbps on the originating interface.
- **Assigned**. The capacity that is assigned to the originating interface. This is the capacity that has been configured at the interface and that can be used for transmit.

Click **Refresh** refresh the displayed data.

Click **Ok** to close the window.

See Also

Configure DTM Interface

Delete Link

The **Delete from Database** operation deletes link symbols from all maps, and its object from the database.

It is only possible for Nimbra Vision to detect whether a link exists or not. If a link cannot be detected, it is because the link either is removed or because it is down. Nimbra Vision does not automatically remove links when they cannot be detected any longer. Instead, the links gets their *linkState* status changed from to *down*, and a critical alarm is raised on the link. You can use this command to remove the links that should not be managed any longer.

If a removed link is detected again, it will re-appear in the DTM Network map, and in the Network database. A link will be detected again if its status is changed, or if the node managed object is refreshed.

See Also

Nimbra Link Properties for details about the link state.

Highlight Link Utilization

You can highlight all the links in maps, where the highlight color depends on the link utilization. To highlight all the links, choose menu
Edit | Highlight Link Utilization. The menu is available when a map is the active frame.

Only links with link state *up* or *noControl* are highlighted.

The link are highlighted using the following colors:

Link Utilization (%)	Color Name	Color
0 - 49	Green	🟩
50 - 74	Yellow	🟨
75 - 100	Orange	🟧

The utilization for a link is determined by the utilization of the interface where the link terminates (property `dtmIfTxCapacityUtilization`, see Nimbra Interface Properties).

Note! *The utilization is not a live value from the node, it is the last known value stored in the database. The Performance Monitoring module periodically updates the utilization.*

© 2003-2010 Net Insight AB, All Rights Reserved

The thresholds values and the colors can be configured in file `<NMS_Home>\conf\clientparameters.conf`.

See Also

Nimbra Interface Properties

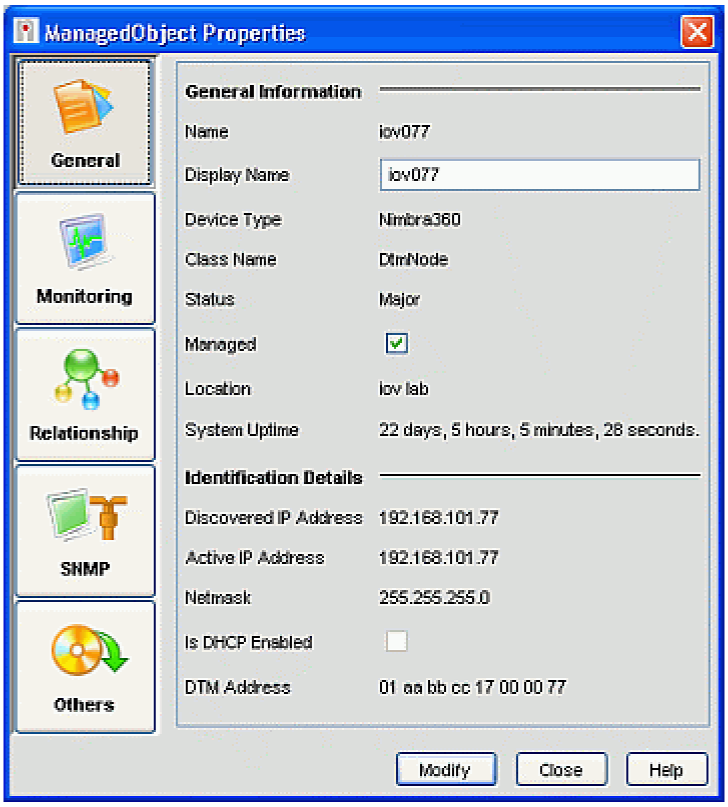
Highlighting in Maps

Device Details

Accessing Device Details

The dialog Managed Properties is a dialog window that displays all the properties about a managed object in the database. The window is generic and its function is the same for all the managed objects regardless of their type or class. Different types of managed objects would contain different types of properties. Please refer to the topic for each type of managed object for details about the different properties (see references below).

To open the Managed Object Properties dialog, select a managed object in a map or in the Network Database, and choose the menu ***Object* | Managed Object Properties**.



General Information	
Name	iov077
Display Name	iov077
Device Type	Nimbra360
Class Name	DtmNode
Status	Major
Managed	✓
Location	iov lab
System Uptime	22 days, 5 hours, 5 minutes, 28 seconds.
Identification Details	
Discovered IP Address	192.168.101.77
Active IP Address	192.168.101.77
Netmask	255.255.255.0
Is DHCP Enabled	☐
DTM Address	01 aa bb cc 17 00 00 77

The Managed Object Properties dialog.

You can edit some of the properties. If you modify the properties, the database will be updated, but change will not propagate to the node. Generally, the dialog is not used for changing the information and the ability to save the edited data should be disabled for most users.

© 2003-2010 Net Insight AB, All Rights Reserved

Refer to the description of the device details for each type of managed object for more information about the properties.

See Also

Map and Device Details in the Web NMS User Guide

Nimbra Node Properties

Nimbra Interface Properties

Nimbra Link Properties

Services Properties

Nimbra Node Properties

The managed object Nimbra Node contains properties for Nimbra nodes. This topic describes all the properties. Some of the properties are for internal use only, but many of the properties might be useful.

Properties are used in Custom Views for e.g. the Network Database or the Custom View maps.

To open a generic form to view and edit the properties, select the DTM node in a map or in the Network Database, and choose the object menu **Nimbra Node | Managed Object Properties**.

Property	Description
activeIpAddress	The IP address that is currently used when reaching the node. If a node has multiple IP addressed, the *activeIpAddress* would be assigned the reachable IP address with the highest priority. If the node has only one IP interface, this is the same as **ipAddress**.
baseMibs	A list of the SNMP MIBs that are implemented by the node, as configured in the file `<NMS_Home>\conf\baseMibs.conf`.
childrenKeys	A list of Managed Objects that are direct children of this node. This is not used for this type of managed object.
classname	Always *DtmNode* for this class of managed objects.
community	The SNMP read community name used when doing an SNMPv1/c2c get operations.
contextName	The SNMPv3 context name. Should normally be the empty string, which is the default context.
currentTimingSource	The name of the interface where the node is receiving its timing (synchronization) information. If the signal is coming from another Nimbra node, then this is the name of the DTM interface.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
currentTimingSourcePeer	The *name* of the Nimbra node connected to the interface *currentTimingSource*. This is the neighboring node from where the timing (synchronization) information is received. If the node is not known, i.e. if the node does not exist in the database, then this is the *nodeId* of the node.
displayName	The name displayed on the map. This value is assigned to the map symbol property **label** when a symbol is created. When the node is discovered and the managed object is created, the value of this is the same as **name**.
dtmAddress	The primary DTM address of the node.
entChangeTime	A time stamp when the inventory information was last updated. This property is updated when Nimbra Vision successfully reads the inventory information from the node. The value is the node's uptime when the inventory information was updated in the node, as presented by the node. The property value is set to 0 when Nimbra Vision fails to read the inventory information. This can happen if e.g. the login information is incorrect. The property does not exist if the node does not support inventory. The entChangeTime is actually read from the ENTITY-MIB::entChangeTime.
eventLogLastChagnedTime	A time stamp when the event log in the node was last changed. This value is updated each time Nimbra Vision gets an indication of that the event log has been updated. The time stamp is time as reported by the node, using its calendar clock. The time stamp is one of the parameters used by the user tester when deciding on if any notifications have been lost. Also, see the NETI-EVENT-MIB.
eventSequenceCounter	The identity of the last received SNMP notification. Each notification is assigned a sequence number by the node, and is increased by one for each notification. The value is one of the parameters used by the user tester when deciding on if any notifications have been lost. Also, see the NETI-EVENT-MIB.
failureCount	The number of consecutive status polls that has failed. The value is reset to 0 when the threshold **failureThreshold** is reached, or the status poll is successful. Also, see **pollFailureCount**.
failureThreshold	The number of consecutive polling failures (**failureCount**) before the objects is declared as failed. When the object is failed, an alarm is raised and the **status** is changed.
groupNames	Not used. Specifies the names of the groups to which the network belongs. Groups can aid in organizing managed objects.
groupMembers	Not used.
hostNetmask	The IP netmask.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
interfaceList	List of all IP interfaces, i.e. all IP addresses that is assigned to the node.
ipAddress	The IP address that was used when the node was discovered/added.
isContainer	Always *false*.
isDHCP	*True* if the node is a DHCP device, otherwise *false*. This should always *false* for Nimbra nodes.
isDtmAccessNode	*True* if the node has at least one access interface, and is thus able to terminate a connection. Otherwise *false*.
isDtmSwitch	*True* if the node has at least two DTM interfaces, and is thus able to switch DTM traffic.
isGroup	Always *false*.
isInterface	Always *false*.
isNetwork	Always *false*.
isNode	Always *true*.
isRouter	*True* if the node routes IP traffic, otherwise *false*. This would be *true* for nodes that routes IP traffic to DLE segments. The node is considered a router if the *ipForwarding* SNMP OID is set to *forwarding*.
isSNMP	Always *true*. The Nimbra node is SNMP enabled.
managed	Whether the managed object is managed or not. When not managed, the status tester will not be executed, and all SNMP notifications from the node represented by the managed object will be ignored, and the status will be set to *unknown*. See Manage and Unmanage a Node for details.
name	Unique name of the managed object. This is the key for the managed object in the database. The name is the host name or DNS name of the node, of if no such name can be found, the IP address. The name of a node can never be changed.
netmask	The IP netmask of the node.
nodeId	A unique identity of the node on the format as a MAC address. The node id `00 00 00 00 00 00` denotes an invalid node id. The value `Duplicate` denotes that more than one node has reported the same node id in a valid format. An empty string denotes that the node does not support reporting of node id.
nodeStartTime	Time stamp of when the node last started. This value is calculated from the current time and the **sysUpTime**. The time is stored as seconds since January 1, 1970 UTC.
parentNet	List of IP network addresses of the networks where the node is located.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
pollFailureCount	The number of consecutive polling failures. The value is reset to 0 when a successful poll has been completed. Compare **failureCount**.
pollInterval	Interval in seconds between polls, for checking the status of the node. You can increase this value to reduce the load on the network. However, this would also mean that it takes longer time to notice if a SNMP notification has been lost. This value is derived from the file `<NMS_Home>\conf\OIDType.data`.
snmpport	The UDP port to use when doing SNMP operations on the node.
state	The updated state of the object. Describes how the data in the managed object is or needs to be updated with the data in the node: *OK* - The managed object is up to date. Its data reflects the data in the node. *Not updated* - An event has been received indicating that data in the node has changed and that the managed object must be updated to reflect this change. The managed object is queued to read the data from the node to get updated. *Update failed* - Indicates failure when attempting to read data from the node when updating the managed object. This state would be set if e.g. there is a communication error while updating the managed object. Will try again. *Sync* - One or multiple events sent from node as traps has been lost. The managed object is queued to restore the lost events from the node. If it turns out that it is not possible to restore the lost events, then a refresh event is generated and the state is changed to *Refresh pending*. *Sync failed* - Indicates failure when attempting to restore lost events, and will try again. This state would be set if e.g. there is a communication error while attempting to read the missing events. *Refresh pending* - A refresh events has been generated because at previous state *Sync* it was determined that a refresh is needed to restore the state. The event filter has not yet processed the event. *Refresh* - The data in the managed object is not up to date with the data in the node, and this is resolved by reading all the data, including services, interfaces, links and alarms, from the node. This state would be set if e.g. the node has restarted, or if the missing events are no longer available in the node. The managed object is scheduled to re-read all data from the node. *Refresh failed* - Indicates a failure when attempting to read data from the node when refreshing the managed object. This state would be set if e.g. there is a communication error while reading the data. Will try again.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	**Description**
status	The highest alarm severity of any alarm associated with this managed object.
statusChangeTime	Time when the alarm **status** was last changed from one status to another, i.e. when the alarm was raised or cleared.
statusPollEnabled	*True* if the status polling has been enabled, otherwise *false*. This would normally be *true*. Status polling is temporarily disabled by Nimbra Vision while it is doing a refresh of the node.
statusUpdateTime	Time when the alarm **status** was last checked.
syncNode	The name of the node where the synchronization originates. This is the root in the synchronization tree within the DTM network.
sysDescr	The system description as specified by *sysDescr* in the SNMP system group. See System Information.
sysLocation	The physical location of the node, as specified by *sysLocation* in the SNMP system group. See System Information.
sysName	The system name as specified by *sysName* in the SNMP system group. See System Information.
sysOID	The system object identified as specified by *sysObjectID* in the SNMP system group. This uniquely identifies the type of node using the SNMP protocol. See the NETI-COMMON-MIB.
sysUpTime	The period that the node has been up and running, as specified by*sysUpTime* in the system SNMP group.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
timeScaleStatus	*uninitiated* - The initial state which indicates that the time of the time scale is not initiated and may not be used for any time keeping. The time scale is however running, so it may be used for relative measures such as those used in round-trip estimates. During the uninitiated state the Time Transfer Control Processing attempts to initiate the TAI/UTC time from either another DTM node in state compensated, or some local UTC source. When such a source is available, it reassigns the time scale and enters the *reassigned* state. *reassigned* - When the time scale of this node have a degenerated time-offset from neighbor nodes being compensated, or a local TAI/UTC source, the node reassigns the time scale and enters the reassigned state, during which the time scale may not be used for precision time-keeping. During the reassigned state the Time Transfer Control Processing attempts to retrace the TAI/UTC time from either another DTM node in state compensated, or some local TAI/UTC source. When the retracing has reduced the time error to within defined limits (1 µs), it enters the *compensated* state. *compensated* - When the time scale of the node has been initiated to TAI/UTC, it can be used for time keeping, but the time error of the time scale is too high (and thus stability and precision is limited). During the compensated state the Time Transfer Control Processing will use the TLL to maintain the TAI/UTC time scale from any neighboring nodes in the *compensated* state. *notSupported* - Time Transfer is disabled or not supported.
tester	The tester that will be used for testing the **status** of the node. For correct behavior on DtmNodes, the tester should be set to *usertest*.
tag	A user defined string that can be used for tagging the managed object. See Tagging Managed Objects.
type	The type of managed object: *NimbraOne*, *Nimbra290*, *Nimbra291*, *Nimbra340*, *Nimbra340HD*, *Nimbra360*, *Nimbra680, Nimbra688* or *NimbraUnknown*, depending on what type node it is.
updateFailureCount	The number of consecutive update operations that have failed when updating information in the managed object from the node. This value is reset to 0 when an update is successful. A topology alarm is raised when an update fails.
uClass	The Java class to run when testing the **status**, if the **tester** is set to *usertest*. For correct behavior on DtmNodes, the value should be *se.netinsight.nv.netwatch.DtmNodeUserTester*.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
userName	The user name to use for secure SNMPv3 communication with the node. This value is used to find the SNMPv3 parameters that shall be used when configuring the device. It is used to look-up the correct security parameters in the SNMPv3 security database (USMTABLE).
version	The SNMP protocol version to use for communicating with the node. This would be *v1* for SNMPv2, *v2* for SNMPv2c and *v3* for SNMPv3.
writeCommunity	The SNMP community name when doing SNMPv1/v2c set operations.

Note! *The property names are case sensitive, and start with a lower case letter*

Nimbra Interface Properties

The Managed Object DTM Interface contains properties for DTM interface. This topic describes all the properties. Some of the properties are for internal use only, but many of the properties might be useful.

Properties are used in Custom Views for e.g. the Network Database or the Custom View maps.

To open a generic form to view and edit the properties, select the DTM interface in the Network Database, and choose the object menu **Interface | Managed Object Properties**.

Property	Description
bundleName	A string that can be used by you. The intention is to use the string to identify bundles in the network. A bundle can be a physical fiber or a fiber bundle capable of carrying multiple DTM links. By naming such a bundle, and assigning that name to the property for all DTM interfaces using the same bundle, it can be easier to detect if e.g. protected connections use different DTM links, that are using the same bundle (and are therefore likely to fail at the same time, in case of e.g. a cable breaking).
childrenKeys	The name of the children. This is the **name** of the DTM link originating on this interface.
classname	Always *DtmInterface* for this class of managed objects.
displayName	The display name. This value is assigned to the map symbol property **label** when a symbol is created.
dtmIfIndex	The index in the SNMP *DtmIfTable* in the NETI-DTM-MIB. This is used for cross-referencing the **name** used by Nimbra Vision, and the index used in SNMP.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
dtmIfMacAddress	The unique id of the DTM interface, in the format of a MAC address. of the DTM interface. The value of `00 00 00 00 00 00` denotes invalid or unknown. The value `Duplicate` denotes that more than one interface has reported the same value in a valid format.
dtmIfName	The name of the DTM interface assigned in the node.
dtmIfOperStatus	The operational state of the DTM interface. See Configure DTM Interface.
dtmIfTxCapacity	The transmit capacity in 512 kbps DTM slots for the interface. This is the total available capacity.
dtmIfTxCapacityUtilization	The used transmit capacity in percent (%) of total transmit capacity. This value is updated when the Performance Monitoring module polls the **dtmIfTxCapacityUtilization** OID from the node. The timestamp in **statusUpdateTime** is updated when this value is updated.
dtmNodeName	The **name** of the node containing the interface.
failureCount	Not used - reserved.
failureThreshold	The number of consecutive polling failures (**failureCount**) before an alarm is raised. Not used.
groupNames	Not used.
isContainer	Always *true*. The interface contains the DTM Links.
isGroup	Always *false*.
isInterface	Always *true*. A DTM Interface is an interface.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
linkClass	The link class controls the behavior of a link when a failure detected on the link or on the node connected to the remote end of the link. The link class decides which values the node status state take. Channels are retained over the link as long as it is not considered *down*. A link is monitored with the following mechanisms: The underlying trunk interface will detect when the signal on the receiving interface disappears or is impossible to use due to errors. This is called a *Signal Failure*. Periodically sending messages, and expecting responses monitor the communication with the node at the remote end of the link. If no response is received within a reasonable time, a failure is assumed. This is called a *Supervision Failure*. The error detection mechanisms can detect failures in one or both directions of a bi-directional link. If a failure is detected in only one direction of a bi-directional link and there is another fully working link in the opposite direction, the remaining working links will all have status *up*. The link classes are: *normal* - If either a Signal Failure or Supervision Failure is detected on the link, the link is considered as *down*. This means that the link is considered *down* as when the link fails. *persistent* - If a Signal Failure is detected on the link, the link is considered as *down*. If a Supervision Failure is detected on the link, the *linkState* becomes *noControl*. This means that the link is not considered *down* as long as it can be verified that it might be possible to send data. *nailed* - If a Signal Failure is detected on the link, the *linkState* for the link is assigned *downKeep*. If a Supervision Failure is detected on the link, the *linkState* for the link is assigned *noControl*. This means that the link is never considered *down*, regardless of any detected faults.
managed	Whether the MO is managed or not. See Manage and Unmanage a Node for details.
name	Unique name of the managed object. This is the key for the managed object in the database. The name is constructed from the DTM interface local name in the node, **DtmIfName**, and the node **name**, separated by an underscore "_".
parentKey	Not used - reserved.
pollInterval	Interval in seconds between polls. Polling is not done on links, so this value is not used.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
state	The updated state of the object. Describes how the data in the managed object is or needs to be updated with the data in the node: *OK* - The managed object is up to date. Its data reflects the data in the node. *Not updated* - An event has been received indicating that data in the node has changed and that the managed object must be updated to reflect this change. The managed object is queued to read the data from the node to get updated. *Update failed* - Indicates failure when attempting to read data from the node when updating the managed object. This state would be set if e.g. there is a communication error while updating the managed object. Will try again.
status	The status of the DTM interface is derived from the operational and administrative state (**dtmIfOperStatus** and **dtmIfAdminStatus**). Note that faults on DTM interfaces are not reported as alarms on the DTM interface managed object, they are reported on the node. This means that the severity of the node is determined by faults on the DTM interface.
statusChangeTime	Time when the alarm **status** was last changed from one status to another, i.e. when the alarm was raised or cleared.
statusPollEnabled	*True* if the status polling has been enabled.
statusUpdateTime	Time when the alarm **status** was last checked.
tag	A user defined string that can be used for tagging the managed object. See Tagging Managed Objects.
tester	Not used - reserved.
type	The type of managed object: *DtmInterface.*
userClass	Not used - reserved.

Note! *The property names are case sensitive, and start with a lower case letter.*

See Also

Channel Persistence

Nimbra Link Properties

The Managed Object DTM Link contains properties for DTM by-pass link connection. This topic describes all the properties. Some of the properties are for internal use only, but many of the properties might be useful.

Properties are used in Custom Views for e.g. the Network Database or the Custom View maps.

To open a generic form to view and edit the properties, select the DTM link in a map or in the Network Database, and choose the object menu **Link | Managed Object Properties**.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
administrativeName	Any string that can be used to label the link. This value is assigned to the map symbol property **label** when a symbol is created. This label can be displayed in the network map by setting the link symbol property **hideTheLinkLabel** to *false*. The intended use is for the user to assign a name.
classname	Always *DtmLink* for this class of managed objects.
displayName	The display name, which are normally not used.
failureCount	Not used - reserved.
groupNames	Not used.
isContainer	Always *false*.
isGroup	Always *false*.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
linkState	The status of the communication with the neighboring node via this link. *up* - the communication with the neighboring node is fully functioning. This is the only state where it is possible to establish new channels over the link. *recover* - the neighboring node is in a recovery state after a node or control function sub-system failure, where it is in the process of recovering the state of all channels. When the process of recovering the channels is completed, the state will change to either *up* if all channels state were successfully recovered, or *limited* if the state of any channels failed to be recovered. *limited* - he neighboring node is not or has not been able to recover the state of all its channels after a node or control function sub-system failure. The already existing channels over the link may be fully functional, but the neighboring node is not accepting any channel signaling. The link must be taken down by the neighboring node for its control function sub-system to be able to assume a known state, which typically requires operator intervention. *noControl* - A valid signal is detected from the neighboring node, but the control function sub-system on the neighboring node is not responding, indicating a remote node or control function sub-system failure. Signaling to the neighboring node is thus not possible. Already existing channels over the link may be fully functional. *downKeep* - The bi-directional communication with the neighboring node is lost. The link is not removed because the interface *linkClass* is configured as *nailed*. *pending* - Indicates that a neighbor has been detected on the receiving part of an interface, but no bi-directional communication has yet been established with the neighboring node. *loopback* - The link is connected from an interface located on the local node. *down* - The link cannot be detected.
managed	Whether the MO is managed or not. See Manage and Unmanage a Node for details.
name	Unique name of the managed object. This is the key for the managed object in the database. The name is constructed from the DTM interfaces separated by the string "-->".
originatingInterface	The **name** of the interface on the originating node.
originatingNode	The **name** of the originating node.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
parentKey	Not used - reserved.
pollInterval	Interval in seconds between polls. Polling is not done on links, so this value is not used.
state	The updated state of the object. Describes how the data in the managed object is or needs to be updated with the data in the node: *OK* - The managed object is up to date. Its data reflects the data in the node. *Not updated* - An event has been received indicating that data in the node has changed and that the managed object must be updated to reflect this change. The managed object is queued to read the data from the node to get updated. *Update failed* - Indicates failure when attempting to read data from the node when updating the managed object. This state would be set if e.g. there is a communication error while updating the managed object. Will try again.
status	The highest alarm severity. The link color on maps will reflect the status of the link. The status is *Critical* if the link is not considered valid (see property **valid**). If the node is unreachable where the link originates, the status is *Warning*.
statusChangeTime	Time when the alarm **status** was last changed from one status to another, i.e. when the alarm was raised or cleared.
statusPollEnabled	*True* if the status polling has been enabled.
statusUpdateTime	Time when the alarm **status** was last checked.
terminatingInterface	The **name** of the interface on the terminating node.
terminatingInterfaceMacAddress	The MAC address of the DTM interface on the terminating node.
terminatingNode	The **name** of the terminating node.
tag	A user defined string that can be used for tagging the managed object. See Tagging Managed Objects.
tester	Not used - reserved.
type	The type of managed object: *DtmLink*.
userClass	Not used - reserved.

Link Status

- **Clear** – the link is up. This means that a link exists between the two interfaces.
- **Warning** – the link status is really unknown, which is because the terminating node cannot be reached to determine the link status.

© 2003-2010 Net Insight AB, All Rights Reserved

- **Minor** -- the link state is either *revover* or *limited.*
- **Major** – the link state is *noControl*, *downKeep* or *loopback*.
- **Critical** – the link state is *down* or *pending*. Note that a link is down also when it no longer exists, i.e. when an earlier detected link cannot be detected anymore. This may be because the link is down due to network fault, or because the network topology has been reconfigured. Links are not automatically removed from the map.

***Note!** The property names are case sensitive, and start with a lower case letter. In the property form, all property names are reformatted and displayed starting with a capital letter, even though the property name starts with a lower case letter.*

See Also

Channel Persistence

IP Network Properties

The managed object Network contains properties for IP networks. This topic describes all the properties. Some of the properties are for internal use only, but many of the properties might be useful.

Properties are used in Custom Views for e.g. the Network Database or the Custom View maps.

To open a generic form to view and edit the properties, select the network in the IP map or in the Network Database, and choose the object menu **Network | Managed Object Properties**.

Property	Description
childrenKeys	A list of managed objects that are direct children of this node. This is always empty for an IP network.
classname	Specifies the class of the managed object. This is always *Network* for this class of managed objects.
contextName	Not used.
discover	*True* if discovery is enabled for the network, otherwise *false*.
discoveryStatus	The status of the discovery of the network. 1. YET_TO_BEGIN - Discovery of network yet to begin. 2. IN_PROGRESS - Discovery if network is in progress. 3. FINISHED - Discovery of the network is finished. 4. DISCOVERY_DISABLED - Takes this value if either **discover** is *false* or **managed** is *false* for this network. You can also see the current discovery in the top right corner of the IP sub-network map windows.
displayName	The name displayed on the map.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
failureCount	The number of consecutive status polls that has failed. The value is reset to 0 when the threshold **failureThreshold** is reached, or the status poll is successful.
failureThreshold	The number of consecutive polling failures (**failureCount**) before the objects is declared as failed. When the object is failed, an alarm is raised and the **status** is changed.
groupNames	Specifies the names of the groups to which the network belongs.
interfaceList	List of all IP interfaces existing in the network.
ipAddress	The IP address of the network.
isContainer	Always *true*. The network contains nodes.
isDHCP	Always *false* for networks.
isGroup	Always *false*.
isInterface	Always *false.*
isNetwork	Always *true*.
isRouter	Always *false*.
managed	Whether the network is managed or not. See Manage and Unmanage a Node for details.
name	Unique name of the managed object. For IP networks, this would be the IP address of the network.
netmask	The IP netmask of the network.
nodeList	List with managed object names of all the nodes in the network.
parentNetmask	Not used.
parentNetwork	Not used.
pollInterval	Interval in seconds between polls, for checking the status of the network.
status	Specifies the status of the network. The status would be the highest alarm severity of any of the nodes in this network.
statusChangeTime	Time when the **status** was last changed from one severity to another.
statusPollEnabled	*True* if the status polling has been enabled, the **status** will be updated according to the **tester**.
statusUpdateTime	Time when the alarm **status** was last checked.
tester	The tester that will be used for testing the **status** of the node This should be *max*. When the tester is *max*, the status of the managed object would reflect the highest severity of any node in the network.
type	The type of managed object. This would always be *Network*.
uClass	The Java class to run when testing the **status**, if the **tester** is set to *usertest*. Because the **tester** should be set to *max*, this value is should be *null*.
userName	Not used

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
version	Not used.

***Note!** The property names are case sensitive, and start with a lower case letter.*

See Also

Network Object Properties in the Web NMS User Guide

Viewing Related Data

View Events Related to Managed Object

To view events related to a managed object: select the managed object in the Network Database or a map, and choose menu **View | Events**. This opens the Event Viewer where only events related to the selected managed object are displayed.

View Alarms Related to Managed Object

To view alarms related to a managed object: select the managed object in the Network Database or a map, and choose menu **View | Alarms**. This opens the Alarm Viewer where only alarms related to the selected managed object are displayed.

View Nimbra Nodes Related to Managed Object

To view Nimbra Nodes (DtmNodes) related to managed objects: select the manage objects in the Network Database or a map, and choose menu **View | Nimbra Node**. This opens the Nimbra Nodes view in the Network Database and displays only the nodes, which is the source node of the selected managed objects. I.e. the node having the selected managed object.

View Nimbra Links Related to Managed Object

To view Nimbra Links (DtmLinks) related to managed objects: select the manage objects in the Network Database or a map, and choose menu **View | Nimbra Links**. This opens the Nimbra Links view in the Network Database and displays only the links, which are terminating on the selected node, the selected interface, or link with same terminating interface as the selected link.

View Nimbra Interfaces Related to Managed Object

To view Nimbra Interfaces (DtmInterfaces) related to managed objects: select the managed objects in the Network Database (or a map), and choose menu **View | Nimbra Interfaces**. This opens the Nimbra Interfaces view in the Network Database and displays only the interfaces, which are on the same nodes as the selected node, or the terminating interface of the selected link, or the selected interface.

View Nimbra Services Related to Managed Objects

To view Nimbra Services (DtmServices) related to managed objects: select the managed objects in the Network Database (or a map), and choose menu **View | Nimbra Services**. This opens the Nimbra Services view in the Network Database and displays only the services, which originates on the selected node, or the selected service.

Note that no services are displayed for selected DtmLink of DtmService. Use the List Channels function to list these services.

© 2003-2010 Net Insight AB, All Rights Reserved

Working with Alarms and Events

Events

An event is generated as a result of that something is happening in the network or in the management system. An SNMP notification is normally converted to an event. If the event represents an alarm (i.e. the event has a severity), the event will result in creation or updating of an alarm.

This chapter describes the following types of events:

- Nimbra Events (common data)
- Nimbra Alarm Event
- Update Event
- Performance Monitor Event (G.826)

Some types of events that are logged are:

- Alarms, generation and clearing of
- Switching of sync source
- Modifications of node configurations (Object modified or created)
- Modification of network topology
- Detection of lost traps
- Node refresh
- Network or node added to or deleted from the database

Nimbra Events

All events generated as a result of a notification (SNMP trap) from the Nimbra network element contains the base properties described in the following table. In addition to these, properties unique for the type of event is also part of the event, as described in additional tables below.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
category	*DTM Node Object Update*. An event indicating that the object **entity** has been updated in the node. *DTM Node Alarms*. The event represents an alarm generated by the Nimbra node. *Topology*. Nimbra Vision detects a modification of the topology database (e.g. a node or network is added or deleted), or modification of the status of how to reach a node. *DTM Node Lost Notification*. Nimbra Vision has detected that at least one notification has been lost from the node. It is trying to recover the lost notifications from the node. *DTM Node Refresh*. Nimbra Vision fails to recover lost notifications from the node, and will refresh the managed object to resolve the problem. *15 minute PM report*. An event holding a G.825 performance report. *24 hour PM report*. An event holding a G.825 performance report.
time (Date/Time)	Time stamp given by Nimbra Vision for the event. This is stored as milliseconds since January 1, 1970.
domain	It is a property in the event object, which holds any domain-specific information based on physical location, functional categorization, or logical categorization of the source of the event. This property is not used by default.
entity (Failure Object)	A unique identifier for the object to which the event belongs. This object is not necessarily the same as a managed object. For failure objects represented within the Nimbra nodes, the format of the entity is: **hostname**/**objectName**/**typeID**/**causeID**. In the case where the event has a severity of *Clear*, *Warning*, *Minor*, *Major*, or *Critical*, the entity is used to correlate the events into alarms. **typeID** and **causeID** are the numerical value of **type** and **cause** as specified in NETI-EVENT-MIB.
groupName	Specifies the group name to which the event belongs. A group name can be used to logically group different events.
id	An index of the event used internally in Nimbra Vision database. The index is unique per event and is assigned when the event is saved to the database after parsers and filters have processed it.
message	A textual message describing the event.
modifiedInNode	The time stamp when for event. The node sets this time stamp using its clock and local time. The time stamp is a string on format *YYYY-M-D,HH:mm:ss.*
network	Information about the network to which the source of the event belongs is specified here. This property is not used by default.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
node	Specifies the node to which the event belongs. In case the event is belonging to a Nimbra node, the property instead contains the **sysName** of the node.
objectName	A human readable name of the object generating the event. This would be the object name assigned by the node. E.g. the name of an interface. Compare **object**.
object	A unique identifier for the object generating the event. For events generated by the Nimbra nodes (**source**), this is the SNMP OID of the object. Some events generated by the nodes are for objects that are not represented via SNMP. In these cases, the value is set to .0.0. Also, compare **objectName**.
eventSequenceCounter	An integer value assigned to the event. Each event sent from the Nimbra node is assigned an integer value in sequence. This value is used in the process to determine if notifications have been lost. The sequencing restarts when at node reboot.
severity	Specifies the severity of the event such as *Clear*, *Warning*, *Minor*, *Major*, *Critical*, *Unreachable* or *Info*. If the event represents an alarm, this would be the severity of the alarm.
source	Specifies the exact source to which the event belongs. This would be the managed object, e.g. the network, node, interface or link to which the event belongs.
tag	The value of tag is inherited from the managed object to which the event pertains. Also, see Tagging Managed Objects.

Nimbra Alarm Event

When an alarm is raised, cleared or otherwise changed on the node, then an Alarm Event is generated. In addition to the base properties described in Nimbra Events above, an Alarm Event has the following properties.

Property	Description
cause	Specifies the probable cause of the event, according to X.733, and is the string exactly as specified in NETI-EVENT-MIB.
purpose	A string assigned to the object in the node. The string describes the purpose of the object.
type	If the event is received from a Nimbra node, this specifies the type of event. If the event is an alarm event, the property is the alarm type according to X.733, and is the string exactly as specified in NETI-EVENT-MIB.

© 2003-2010 Net Insight AB, All Rights Reserved

Update Event

When an object is modified in the node, then an Updated Event is generated. In addition the base properties described in Nimbra Events above, an Update Event has the following properties.

Property	Description
eventType	This described the type of updated event. An updated event is sent by the Nimbra node if something is changed on the configuration on the **object**. The value can be *none*, *create*, *modify* or *delete*
previousEventLogLastChangeTime	The time stamp of the last event sent by the node. This timestamp is given by the node.

Performance Monitor Event (G.826)

If the Nimbra node is configured to periodically send G.826 performance monitoring events, such events are generated. In addition to the base properties as described in Nimbra Events above, a Performance Monitoring Event has the properties as below. The properties are according to G.826.

Property	Description
BBE	Background Block Errors. Counts number of error blocks found during one second of available time that is not part of SES.
ES	Errored Seconds. Counts how many seconds of available time that has at least one errored block (BBE).
IS	Is Suspect, either *true* of *false*. If *true*, then it gives an indication that the counter values are not correct.
SES	Severely Errored Seconds. Counts how many seconds of available time that have been seriously faulty. One second of available time containing ≥ 30% EB or at least one defect. Ten consecutive SES will begin UAT while ten consecutive non-SES will end UAT. SES is a subset of ES.
UAS	Unavailable Seconds. Counts how many seconds a SAP (service access point) has been unavailable during it's up time. One second of UAT. During UAT calculations of BBE, ES and SES are inhibited and no data collection of them is performed.
ZS	Zeroed Suppression Count. Number of faultless periods before this period.
SS	Slip Second, one second containing one or more Slips. Applicable for the trunk modules.

See Also

Fault Details in the Web NMS Users Guide

© 2003-2010 Net Insight AB, All Rights Reserved

Archived Events

Events are saved into an events archive in the database. The purpose of the archive is to be able to keep a large number of events over a long period of time. The function complements the Networks Events view, which provides a fast access to, and keeps a live update of the latest number of events. The limit of the number of events is set by the used database. As of MySQL 5.0, the limit is 2^{32} (~4.3 x 10^{9}) rows. This can be compared to the 40,000 (default value) actively managed number of events in the Network Events view.

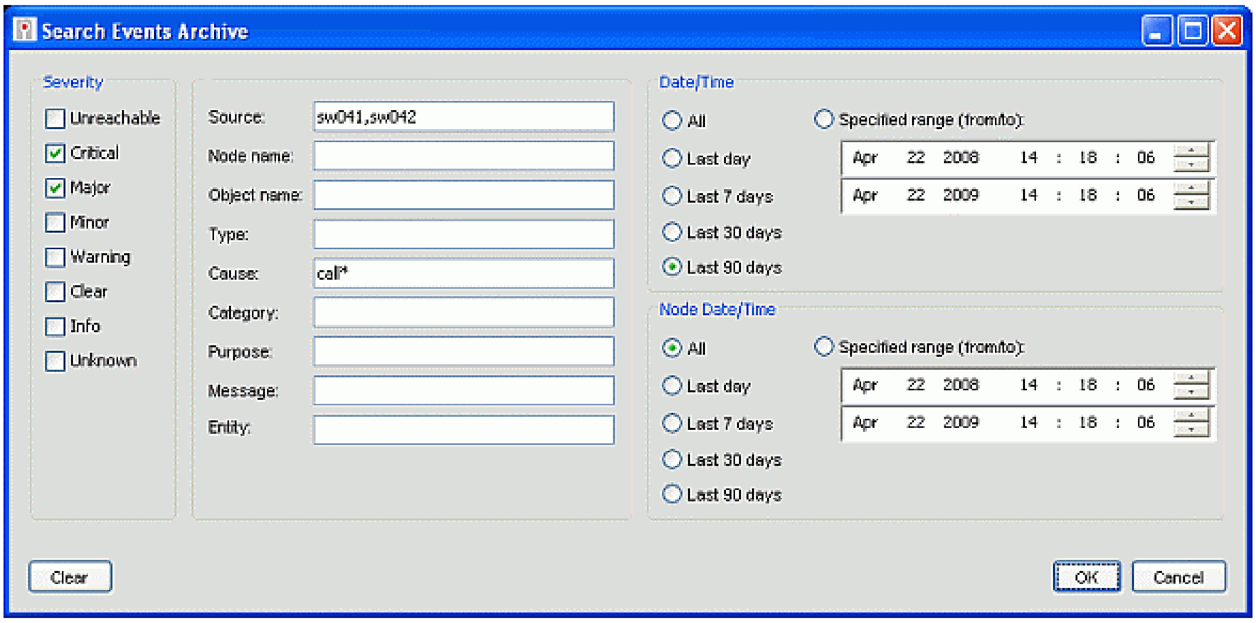
An archived event contains a sub-set of the properties of the events. The following properties are included in the archived event:

- severity
- source
- tag
- node
- objectName (Object)
- cause
- type
- category
- purpose
- modifiedInNode (Node Date/Time)
- time (Date/Time)
- message (text)
- entity (Failure Object)

See Events for description of the different properties.

To access the archived events, open the **Archived Events** node under the **Fault Management** node in the Tree. This opens a panel that contains a table of archived events. To view some events, use menu **Edit | Search** (or use toolbar button), which opens a search dialog. Enter the search criteria and click button **OK**. The search criteria can use wildcards etc., see Writing Filer Criteria for description on how to write search criteria. The Archived Events panel will be updated with events matching the entered criteria. The table is limited to 5000 rows.

© 2003-2010 Net Insight AB, All Rights Reserved


Search Events Archive

Severity

- [ ] Unreachable
- [✓] Critical
- [✓] Major
- [ ] Minor
- [ ] Warning
- [ ] Clear
- [ ] Info
- [ ] Unknown

Source:	sw041,sw042
Node name:	
Object name:	
Type:	
Cause:	call*
Category:	
Purpose:	
Message:	
Entity:	

Date/Time

All · Last day · Last 7 days · Last 30 days · [✓ Last 90 days] · Specified range (from/to):

Apr 22 2008 14 : 18 : 06
Apr 22 2009 14 : 18 : 06

Node Date/Time

[✓ All] · Last day · Last 7 days · Last 30 days · Last 90 days · Specified range (from/to):

Apr 22 2008 14 : 18 : 06
Apr 22 2009 14 : 18 : 06

Dialog for searching in Events Archive.

Events are added to the archive by an event filter. See Processing Events and Alarms.

A user is restricted to view only archived events as specified by the Custom View Scope named **Events**.

See Also

Events

Cleanup Archived Events

Processing Events and Alarms

Writing Filter Criteria

Alarms

An alarm is generated when a fault or failure is detected. Most of the alarms in a Nimbra network is detected and reported by the node. The alarm database does thus become a collection of all the alarms from all the managed nodes.

In addition to the alarms generated by the nodes, Nimbra Vision can generate alarms when it detects failures. This would typically be Typology failures generated when Nimbra Vision fails to connect to a managed object. It is also possible to monitor individual SNMP objects, and generate an event when its value crosses a threshold.

Alarms are built up from events. From an alarm, you can see all the events that have affected the status of the alarm.

The alarms generated by the node conform to X.733 standard. These alarms contain the following properties:

Property	Description
category	Specifies the category to which the alarm belongs. This is always *DTM Node Alarms* for alarms generated by the node. For topology related alarms, the value is *Topology*.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
cause	Specifies the probable cause of the alarm, according to X.733. See NETI-EVENT-MIB.
createTime	Time stamp when the alarm was first created. This would be the time stamp for the first non-clear event on the entity, which would create the alarm. Nimbra Vision server creates the time stamp. The time stamp is stored as milliseconds since January 1, 1970 UTC.
entity (Failuire Object)	A unique identifier for the object to which the alarm belongs. This object is not necessarily the same as a managed object. For failure objects represented within the Nimbra nodes, the format of the identifier is: **hostname**/**objectName**/**typeID**/**causeID**. The value is used when correlating events into alarms, where all events with the same entity correlate to the same alarm. **typeID** and **causeID** are the numerical values of the **type** and **cause** as specified in NETI-EVENT-MIB.
groupName	The group to which the alarm belong. A group can be used to logically group different alarms.
id	The ID of the last event that modified the alarm. This is an integer value.
message	The alarm message; a description of the alarm.
modifiedInNode	The time stamp when the alarm status was last modified. The node sets this time stamp. Compare **modTime**. The time stamp is stored as a string on format YYYY-M-D,HH:mm:ss.
modTime	Time stamp when the alarm was last modified. Nimbra Vision server sets this time stamp. Compare **modifiedInNode**. The time stamp is stored as milliseconds since January 1, 1970 UTC.
object	The SNMP OID on the node **source** representing the alarm. Some events generated by the nodes are for objects that are not represented via SNMP. In these cases, the value is set to .0.0. Compare **failureObject** and **objectName**.
objectName	A human readable name of the alarming object. This would be the object name assigned by the node. Compare **object** and **failureObject**.
previousSeverity	The previous status.
purpose	A string assigned to the object in the node. The string describes the purpose of the object.
severity	The status of the alarm as an integer value: 6, 5, 4, 3, 2 or 1 representing *Clear*, *Warning*, *Minor*, *Major*, *Critical* or *Unreachable*.
source	Specifies the exact source to which the alarm belongs. This would be the managed object, e.g. the network, node, interface or link to which the alarm belongs.
tag	Updated from the tag on last event that updates the alert. Also, see Tagging Managed Objects.
type	Type of alarm, according to X.733. See NETI-EVENT-MIB.
who (Owner)	The user name of the person that has picked-up the alarm. This person is considered responsible for the actions resulted from the alarm.

© 2003-2010 Net Insight AB, All Rights Reserved

See Also

Fault Details in the Web NMS Users Guide

Processing Events and Alarms

Nimbra Vision relies on traps from the Nimbra network elements to update the managed objects and alerts. This topic briefly describes how Nimbra Vision processed the traps and resulting events and alarms. The trap/event/alarm are processed in a pipeline where the traps/events/alarms are processed in the order they are entering the pipeline.

7. The trap parser receives the trap from the Nimbra network element. If the trap parser is choked, then the trap is temporarily stored on disk until the trap parser is able to process traps.
8. The trap parser checks the event sequence counter and the timestamp in the received trap.
    a. If the event sequence counter and timestamp are as expected, then the trap is forwarded as an UPDATE event or an ALARM event.
    b. If the trap parser detects lost traps, it forwards a SYNC event.
    c. If the trap parser detects that the node has restart, it forwards a REFRESH event.
    d. If the managed object representing the network element is Unmanaged, then the trap is discarded.
9. The event parsers receive the forwarded events. An event can reformat the contents in the event, i.e. modify the event properties. The event parser has a criteria definition, and if the event matches the criteria, then the event is modified as configured in the event parser. The event parser forwards the (possibly modified) event to the next event parser. The last event parser forwards the events to the event filters. Nimbra Vision comes with the following event parsers:
    a. *15min PerfMon alarms*. Sets the failure object so that it is possible to distinguish 15 minute performance monitoring alarms from the 24 hour performance monitoring.
    b. *24hr PerMon alarms*. Sets the failure object so that it is possible to distinguish24 hour performance monitoring alarms from the 15 minutes performance monitoring.
10. The event filter receives the forwarded events form the last event parsers. The purpose of an event filter is to execute actions. The event filter has a criteria definition, and if the event matches the criteria, then the actions defined in the event filter are executed. You can add your own event filters to execute actions when certain events are received. Nimbra Vision comes with the following event filters necessary for its operation:
    a. *Process DtmNode Event*. This event filter processes the UPDATE, SYNC and REFRESH events. It sets the state of the corresponding managed object in the database for update, synchronization or refresh, but no further action is then taken here.
    b. *Save Event to Archive*. This event filter saves the event into the event archive. If a match criteria is added to the event filter, then only the events matching the criteria is added.

© 2003-2010 Net Insight AB, All Rights Reserved

c. *Change to PerfMon Event*. If the event is a result of a G.826 performance report trap sent from the Nimbra network element, then this filter changes the event to an event with dedicated performance data properties.

d. *Save PerfMon Event to Archive*. This event filter adds events that are representing a G.826 performance monitoring report to the PM report archive. If a criteria is added to the event filter, then only reports matching the criteria is added to the archive.

e. *Update Event Sequence Counter*. This filter updates the event sequence counter property (*eventSequenceCounter*) in the DtmNode managed object, marking that the process of this event is complete. This filter is not marking this for ALARM events (that is done later in an alarm filter). This filter shall be placed after the other event filters to ensure that marking is done when the event has been processed by all filters.

11. When the last event filter has processed the event, the event is stored into the database, and gets a unique id assigned.

12. If the event has a severity (e.g. Clear, Warning, Minor, Major, Critical, Unreachable), then it is considered and alarm, and is correlated with the existing alarm as: if an alarm already exist on the entity (failure object), then the that alarm is updated and forwarded to the alarm filters. Otherwise a new alarm is created based on the event and forwarded to the alarm filters.

13. The alarm filter receives the forwarded alarms. The purpose of an alarm filter is to execute actions. Just like the event filters, then alarm filter has a criteria definition and associated actions. You can add your own alarm filters to execute actions when certain alarms are received. Nimbra Vision comes with the following alarm filters necessary for its operation:

    a. *Redundant Headends*. This filter is part of the Redundant Headend functionality. It makes the function detect and react on alarms.

    b. *Preemption*. This filter is part of the Preemption functionality. It makes the function detect and react on alarms.

    c. *Update Event Sequence Counter*. . This filter updates the event sequence counter property (*eventSequenceCounter*) in the DtmNode managed object, marking that the process of this alarm is complete. This filter shall be placed after the other alarm filters to ensure that marking is done when the alarm has been processed by all filters.

14. When the last alarm filter has processed the alarm, the alarm is stored in the database.

Filter Type	**File Name**	**Custom Filter Class Name**
Event	Process DtmNodeEvent	`se.netinsight.nv.event.DtmNodeEventFilter`
Event	Add Purpose to Event	`se.netinsight.nv.event.AddPurposeEventFilter`
Event	Save Event to Archive	`se.netinsight.nv.event.ArchiveEventEventFilter`
Event	Change to	`se.netinsight.nv.event.`

© 2003-2010 Net Insight AB, All Rights Reserved

Filter Type	File Name	Custom Filter Class Name
	PerfMon Event	`PerfDataEventFilter`
Event	Save PerfMon Event to Archive	`se.netinsight.nv.event.PerfMonEventFilter`
Event	Update Event Sequence Counter	`se.netinsight.nv.event.UpdateEventSequenceCounterFilter`
Alarm	Redundant Headends	`se.netinsight.nv.headend.RedundantHeadendsAlarmFilter`
Alarm	Preemption	`se.netinsight.nv.preemption.PreemptionAlarmFilter`
Alarm	Update Event Sequence Counter	`se.netinsight.nv.event.UpdateEventSequenceCounterFilter`

Viewing Related Data

View Events Related to Alarms

To view alarms related to events: select the events in the Event Viewer, and choose menu **View | Alarms**. This opens the Alarm Viewer where only the alarms that are a result of the selected events are displayed.

View Alarms Related to Events

To view events related to alarms: select the alarms in the Alarm Viewer, and choose menu **View | Events**. This opens the Event Viewer where only the events related to the alarms are displayed.

View Managed Objects Related to Events or Alarms

To view managed objects (such as nodes, services, interfaces or links) that are related to events or alarms: select the events or alarms in the Event Viewer or Alarm Viewer, and choose menu **View | Managed Object**. This opens an applicable custom view in the Network Database where only the managed objects related to the selected events or alarms are displayed. If multiple types of managed objects are listed, then the general custom view Network Database is displayed, otherwise the custom view dedicated for the type of managed object.

Working with Policies

Introduction to Policies

If you want Nimbra Vision to perform or execute actions, you can use policies. A policy is a definition of an action that can be executed by Nimbra Vision. A policy may be executed periodically, according to a schedule, or manually.

Typically, policies are used to generate reports, create backups or clean-up the database.

© 2003-2010 Net Insight AB, All Rights Reserved

There are two types of policies. Periodic policies are executed in time intervals. The time interval is configured per policy. Scheduled policies are executed according to a schedule. The schedule specifies the day/date and time when the policy shall be executed.

To access the policies, select the **Policies** under the **Administration Tools** in the Tree.

Operations on Policies

- To add a new policy, choose menu **Policy | Add Policy**. This will open up a dialog where you can select the type of policy, and specify the name of the policy. Once the policy has been created, the name cannot be changed.
- To modify an existing policy, select the policy and choose menu **Edit | Update Policy**. This will open the policy property editor.
- To manually execute a policy, select the policy and choose menu **Edit | Execute Policy**. This will trigger the policy to be executed. You can execute a policy regardless if it is scheduled or not. However, the policy must be enabled. A periodic policy is executing until it is stopped, performing its task periodically.
- To stop execution of a policy, select the policy and choose the menu **Edit | Stop Policy**. Only an executing policy can be stopped.
- To edit the schedule for a scheduled policy, select the policy and choose the menu **Edit | Schedule Policy**. This will open the policy schedule editor. The schedule editor can also be opened from the policy property editor.
- To enable or disable a policy, edit the policy and set the **status** to Enable or Disable. A disabled policy cannot be executed.
- To delete an existing policy, select the policy and choose menu **Edit | Delete Policy**.

Existing Policies

The following policies exists:

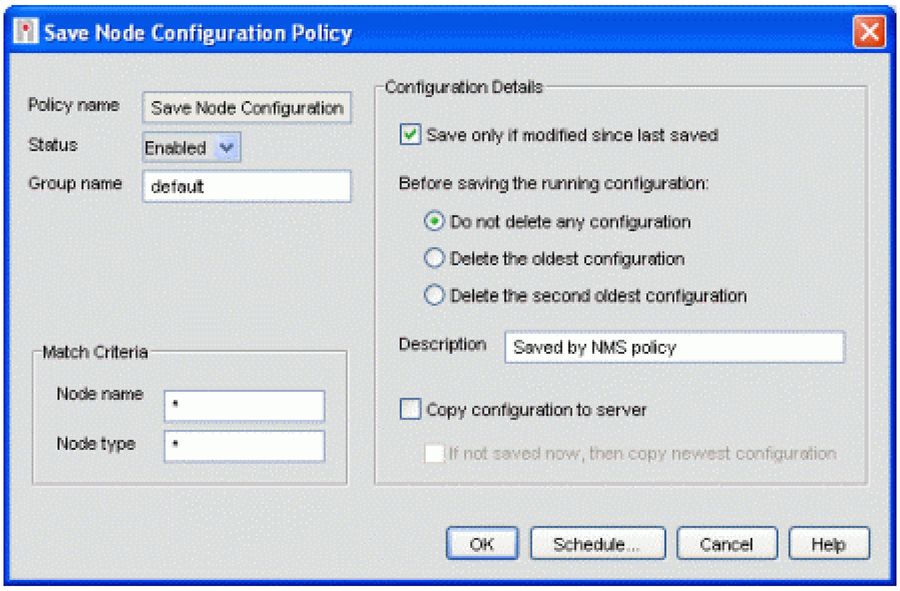
- Save DTM Node Registry. Saves the current configuration in DTM nodes.
- Generate Reports. Generates Reporter Reports for Nimbra Interfaces and their utilization.
- Host List Push. Updates list of DTM host names to DTM addresses.
- Statistics Table Cleanup. Purge old statistics collection.
- Event to Log File. Copy events to log file.
- Database Backup. Backup the database.
- Alert Escalation. Escalate alarms if they keep their status too long.
- Export Inventory. Search Inventory database and exports result to file.
- Cleanup Archived PM Reports. Remove old archived PM reports.
- Cleanup Archived Events. Remove old archived events.

© 2003-2010 Net Insight AB, All Rights Reserved

Save DTM Node Registry

The Save DTM Node Registry Policy saves the current configuration locally on each node for a set of Nimbra nodes. In addition, it also optionally copies the saved configuration to Nimbra Vision server. If there is no space available on the node to save the configuration, the running configuration will not be saved on that node.

The policy is executed for all the Nimbra nodes that matches the criteria Type and Node name. The function is similar to the function Save Local Configuration on Multiple Nodes.


Save Node Configuration Policy

Policy name	Save Node Configuration
Status	Enabled
Group name	default

Match Criteria

Node name	*
Node type	*

Configuration Details

[✓] Save only if modified since last saved

Before saving the running configuration:

[● Do not delete any configuration]
○ Delete the oldest configuration
○ Delete the second oldest configuration

Description: Saved by NMS policy

[ ] Copy configuration to server

[ ] If not saved now, then copy newest configuration

Save node configuration policy.

The policy form contains the following:

- **Policy name** is the name of the instance of the policy. This identifies the policy. The name is set when you create the policy. It cannot be changed after the policy has been created.
- **Status**. The policy can be enabled or disabled. A policy cannot be executed when it is disabled.
- **Group name**. The group name of the policy. A group name can be used to administratively group different policies. This can make it easier to find and filter policies if you have many policies defined.
- **Node name**. This is a criterion. Only the Nimbra nodes with a name matching this will have its running configuration saved
- **Node type**. This is a criterion. Only the Nimbra nodes with a type matching this will have its running configuration saved.
- **Save only if modified since last saved**. The function checks if the current configuration in the node has changed since it was last saved. Note that the configuration may have been saved by some other function. If it has changed, the configuration will be saved. If it has not changed, the configuration is not saved.
- **Description**. A description of the configuration. When the configuration is saved, this data will be used.
- **Before saving the running configuration:** This specifies if a previous saved configuration shall be deleted prior saving the current

© 2003-2010 Net Insight AB, All Rights Reserved

configuration, and in that case ensures that there will be enough space on the node to save the running configuration:

- **Do not delete any configuration** indicates to not delete any configuration on the node. If there are no space left on the node to save the running configuration, this policy will fail for that node only.
- **Delete the oldest configuration** indicates to deletes the oldest configuration on the node prior saving the current configuration.
- **Delete the second oldest configuration** indicates to delete the second oldest saved configuration on the node prior saving the current configuration.

- **Copy configuration to server**. If this checkbox is checked, then after the configuration has been saved, it will be copied to the server. The default location is `<NMS_Home>\backup\nodeconfigurations`. The location can be configured in the file `<NMS_Home>\conf\serverparameters.conf`. The policy is using FTP to copy the configuration to the server. It will use the login credentials as configured in the function Node Users.
- **If not saved now, then copy newest configuration**. If this checkbox is checked, then if a configuration is not saved when the policy is executed, then the newest existing configuration will be copied to the server instead. The configuration will not be copied if a copy with the same name and time stamp already exists on the server for the node. Enabling this function ensures that a copy of the latest configuration will always exist on the server, regardless of how it has been saved on the node, if at all.

This is a scheduled policy. Select the button **Schedule** to open a form to schedule the policy.

The result of running the policy is logged in the Audit Log, which can be accessed from the Tree node **Configuration | Audit**.

Nimbra Vision does not include function to restoring a saved configuration from the Nimbra Vision server to the node. This function is available in the Nimbra node's element manager. Please consult its documentation for details.

See Also

Managing Local Configuration Files

Save Local Configuration on Multiple Nodes

Writing Filter Criteria

Node Users

Generate Reports

The Generate Reports policy generates daily, weekly and monthly reports from statistics collected from DTM interfaces. The reports are web pages that can be accessed from a web browser.

The report generation requires that there are statistical data collected for used transmit capacity on the DTM interfaces. The name of the statistics is by default **Trunk Tx (slots)**. This may be configured in the file `<NMS_Home>\conf\serverparameters.conf`.

© 2003-2010 Net Insight AB, All Rights Reserved

To access the generated reports, first select **Configured Collection** under **Performance** in the Tree. Then, choose menu **View | Reporter Reports**. This opens a overview page from where all the reports can be accessed.

The policy form contains the following:

- **Policy name** is the name of the instance of the policy. This identifies the policy. The name is set when you create the policy. It cannot be changed when the policy has been created.
- **Group name**. The group name of the policy. A group name can be used to administratively group different policies. This can make it easier to find and filter policies if you have many policies defined.
- **Status**. The policy can be enabled or disabled. A policy cannot be executed when it is disabled.
- **Midnight Report**. Check this to generate reports for a 24 hr period ending last midnight from the scheduled time.
- **Daily Report**. Check this to generate reports for a 24 hr period, ending at the last whole hour from the scheduled time.
- **Weekly Report**. Check this to generate reports for a 7-day period ending last midnight from the scheduled time.
- **Monthly Report**. Check this to generate reports for a complete month, ending last midnight from the scheduled time.

This is a scheduled policy. Select the button **Schedule** to open a form to schedule the policy.

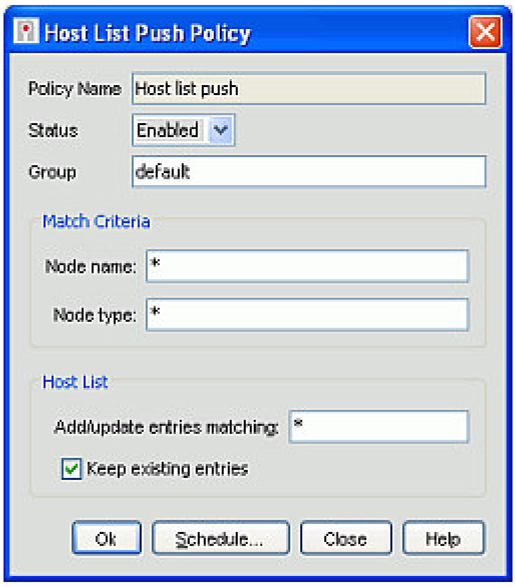
The reports are stored on the Nimbra Vision server file system in the folder *<NMS_HOME>*`\reports\dtmIfReports`. You need to delete old reports periodically to ensure that the disk is not filled up.

The Table of Content web page is re-generated each time the policy is executed.

Host List Push

The Push Host List policy updates the DTM host list in the Nimbra nodes. It pushes the information about host name and DTM addresses as known by Nimbra Vision to a set of Nimbra nodes. The host list contains host name to DTM address mappings. These mappings can be used locally within the node when addressing other nodes.

© 2003-2010 Net Insight AB, All Rights Reserved


Host List Push Policy

Policy Name: Host list push

Status: Enabled

Group: default

Match Criteria

Node name: *

Node type: *

Host List

Add/update entries matching: *

[✓] Keep existing entries

Configuration dialog for the Host list push policy

The policy compiles a host list based on Nimbra nodes known by Nimbra Vision, and update nodes with this list. The policy form contains the following:

- **Policy name** is the name of the instance of the policy. This identifies the policy. The name is set when you create the policy. It cannot be changed after the policy has been created.
- **Status**. The policy can be enabled or disabled. A policy cannot be executed when it is disabled.
- **Group name**. The group name of the policy. A group name can be used to administratively group different policies. This can make it easier to find and filter policies if you have many policies defined.

Deciding what nodes to update:

- **Node name**. This is a criterion. Only the Nimbra nodes with a name matching this will have its host list updated.
- **Node type**. This is a criterion. Only the Nimbra nodes with a type matching this will have its host list updated.

Compiling the host name list:

- **Add/updated entries matching**. This is a criterion for how to compile the host list. When the policy is executed, all the names of all Nimbra nodes as known by Nimbra Vision (i.e. all the Nimbra node managed objects in the Nimbra Vision database) are checked against this criterion. If a name is matched, it is included in the list together with its DTM address. The DTM address is retrieved from the corresponding managed object in the database. This compiled list is then used when updating the nodes' existing lists.
- **Keep existing entries**. This checkbox controls how entries in the node's already existing list are affected when the policy is executed. If this checkbox is checked, any entry for host names that already exist in the node's list and that does not exist in the compiled list, are retained. This means that entries are added and updated according to the compiled list, but never removed. If the checkbox is cleared, then the node's list are replaced with the compiled list. This means that the resulting list in the nodes will be an exact copy of the compiled list.

Note! *When the list is compiled, any dot (".") in the node name is replaced by an underscore ("_"). I.e. "*`node.domain`*" is replaced by "**node_domain**".*

© 2003-2010 Net Insight AB, All Rights Reserved

The result of running the policy is logged in the Audit Log, which can be accessed from the Tree node **Configuration | Audit**.

See Also

Writing Filter Criteria

Statistics Table Cleanup

The Statistics Table Cleanup policy removes old statistics from the database. To avoid filling the database with statistical data, you have to periodically remove old statistical data. This is a periodic policy.

When the policy is executed, it will check if the current hour of the day is Cleanup Hour. If so, the database is cleaned.

The policy form contains the following:

- **Policy name** is the name of the instance of the policy. This identifies the policy. The name is set when you create the policy. It cannot be changed when the policy has been created.
- **Status**. The policy can be enabled or disabled. A policy cannot be executed when it is disabled.
- **Group name**. The group name of the policy. A group name can be used to administratively group different policies. This can make it easier to find and filter policies if you have many policies defined.
- **Cleanup hour**. This determines when to clean up the statistics (hour of the day). It can happen at any time during the hour, and the time in hour cannot be controlled. The default value is 0, i.e., done between 12 at midnight and 1 a.m.
- **Delete data older than (days)**. This determines how long to store the data in the database before it is deleted. Data tables older that this number of days will be deleted. The default value is seven days.
- **Database table**. This is the name of the table that stores the statistical data. By default, the name is `STATSDATA%`. This table is also the default table for collected statistical data. Unless you have changed the data collection parameters, you should specify this table. If you have specified your own table name in the data collection parameters, that table name should be specified in this field. In reality, any table that stores the time in the time field can be specified. However, this is intended only for the statistics tables

Events to Log File

The Event to Log File policy logs events generated during a particular period to a file.

This is a periodic policy. All events generated since the last time the policy was executed is read from the event database and are stored in a file. For example, if 1000 events are read during a certain period, the next time, when the policy is executed, the next set of events are read (from the 1001st event).

Multiple instances of the policy will maintain its own identity of the last event that was saved the previous period.

The policy does not delete any events from the database.

© 2003-2010 Net Insight AB, All Rights Reserved

The policy form contains the following:

- **Policy name** is the name of the instance of the policy. This identifies the policy. The name is set when you create the policy. It cannot be changed when the policy has been created.
- **Status**. The policy can be enabled or disabled. A policy cannot be executed when it is disabled.
- **Group name**. The group name of the policy. A group name can be used to administratively group different policies. This can make it easier to find and filter policies if you have many policies defined.
- **Period (hours)**. The period over which the policy is executed. By default, the period is set to one day, i..e. 24 hours.
- **Save in folder**. The directory (folder) on the server where the log files with stored events shall be saved. The directory must already exist. You can provide a relative path name (e.g. `event`) or an absolute path name (e.g. `C:\event`). The directory is not automatically created, so you have to create the directory.

NMS Backup

The NMS backup policy takes a backup of the Nimbra Vision database.

- **Policy name** is the name of the instance of the policy. This identifies the policy. The name is set when you create the policy. It cannot be changed when the policy has been created.
- **Status**. The policy can be enabled or disabled. A policy cannot be executed when it is disabled.
- **Group name**. The group name of the policy. A group name can be used to administratively group different policies. This can make it easier to find and filter policies if you have many policies defined.
- **Backup classes**. In this field, you have to enter the class name implementing ***com.adventnet.nms.startnms.BackUpInterface*** with the package structure. Multiple class names can also be given, separated with commas. The default class **jdbc.BackIpImpl** makes a backup of the database and stored the backup on the server in the folder *<NMS_Home>*`\backup`.

This is a scheduled policy. Select the button **Schedule** to open a form to schedule the policy.

See Also

Restore

Alert Escalation

The Alert Escalation policy looks at all the Alarms in the database and checks whether any Alarm is in the same state (without any change in severity) for a specified period of time (this time value is specified by the user). If yes, it takes the specified action (configured by the user).

You can specify criteria for which alarms to consider.

© 2003-2010 Net Insight AB, All Rights Reserved

This is a periodic policy. Each time the policy is executed, the alarms matching the criteria are considered. Note that an alarm, which has not had its severity changed, will match equally every time the policy is executed, and the specified action will be taken each time.

You can specify the following type of actions:

- Suppress alarm
- Send SNMPv1 or SNMPv2c trap
- Custom filter, i.e. execute any Java class
- Run command, i.e. execute any program on the server
- Send e-mail
- Set severity, i.e. specify a new severity for the alarm.

See Also

Writing Filter Criteria

Alert or Escalation Policy in the Web NSM Administrator Guide

Export Inventory

The Export Inventory policy exports data about inventory to a file on the server's file system. For each inventory object in the database that matches the Search Criteria, a record is put to the output file when the policy is being executed. A record consists of a row and contains one column for every inventory object property. The record contains all the inventory object properties, regardless if the property has a value or not. If the property has no value, then the column is empty. The columns are ordered alphabetically based on the column name (i.e. property name). See Inventory Properties for a list and description of properties.

The policy form contains the following:

- **Policy name** is the name of the instance of the policy. This identifies the policy. The name is set when you create the policy. It cannot be changed after the policy has been created.
- **Status**. The policy can be enabled or disabled. A policy cannot be executed when it is disabled.
- **Group name**. The group name of the policy. A group name can be used to administratively group different policies. This can make it easier to find and filter policies if you have many policies defined.
- **Export to file**. Path to the export file on the Nimbra Vision server. If this is a relative file name, then it is relative to the Nimbra Vision server installation folder.
- **Column separator**. A string that will separate each column. Typically, this would be a tab character or semi-colon. Tab can be expressed with `\t`.
- **Show header**. When checkbox is selected, then the first row in the report will include the column names.
- **Search Criteria**. Only records matching the criteria is exported to the file. See Inventory Properties for description on the different properties. See Writing Filer Criteria on how to write criteria.

© 2003-2010 Net Insight AB, All Rights Reserved

See Also

Inventory Properties

Writing Filter Criteria

Cleanup Archived PM Reports

The Cleanup Archived PM Reports policy removes old PM reports stored in the Archived PM Reports. To avoid filling the database with too much data, old archived PM reports should be removed from the database. The database is able to hold a large number of reports. The limit of the number of reports depends on the used database, and on available disk storage.

- **Policy name** is the name of the instance of the policy. This identifies the policy. The name is set when you create the policy. It cannot be changed after the policy has been created.
- **Status**. The policy can be enabled or disabled. A policy cannot be executed when it is disabled.
- **Group name**. The group name of the policy. A group name can be used to administratively group different policies. This can make it easier to find and filter policies if you have many policies defined.
- **Delete reports older than (days)**. Reports older than the specified number of days are removed.

This is a scheduled policy. Select the button **Schedule** to open a form to schedule the policy.

See Also

Archived PM Reports

Cleanup Archived Events

The Cleanup Archived Events policy removes old events stored in the Archived Events. To avoid filling the database with too much data, old archived events should be removed from the database. The database is able to hold a large number of events. The limit of the number of events depends on the used database, and on available disk storage.

- **Policy name** is the name of the instance of the policy. This identifies the policy. The name is set when you create the policy. It cannot be changed after the policy has been created.
- **Status**. The policy can be enabled or disabled. A policy cannot be executed when it is disabled.
- **Group name**. The group name of the policy. A group name can be used to administratively group different policies. This can make it easier to find and filter policies if you have many policies defined.
- **Delete reports older than (days)**. Events older than the specified number of days are removed.

© 2003-2010 Net Insight AB, All Rights Reserved

This is a scheduled policy. Select the button **Schedule** to open a form to schedule the policy.

See Also

Archived Events

Performance Monitoring

Introduction to Performance Monitoring

Performance Monitoring is the task when the operation of the network and its nodes are monitored, and when the data is analyzed. Monitoring the network is essential to get early warning on performance degradation, high utilization or other factors. The performance data can also be used when troubleshooting.

Nimbra Vision has two types of performance data:

- Collected statistics, where Nimbra Vision actively periodically polls the nodes for data and stores the data in its database. The performance module is responsible for this.
- G.826 type data. These are performance reports generated by the nodes every 15 minutes and 24 hours. The nodes send the information to Nimbra Vision as events and are stored in Nimbra Vision as such. See Events for details about these performance events. The G.826 performance reports are listed in Custom Views, just as any other event.

To access the performance data, select the node **Performance** in the tree.

The performance module is divided into two parts.

- Collecting statistics. Data is periodically collected from the nodes.
- Reports. The collected data can be viewed in real-time, as historical data, or as reports where data is also further processed and summarized.

To view and configure the collection of performance data, select the node **Performance | Configured Collection** from the Tree.

An easy way to view all the statistics for one node or some nodes is to select the node(s) in the map or the Network Database, and choose menu **View | Statistics**. This will open the **Configured Collection** under **Performance** in the Tree, with statistical collections for the selected nodes only.

For each Nimbra node that is discovered, a set of statistics collections is generated. A statistics collection is a definition of data that shall be collected, i.e. what SNMP variable, and when. Polling objects create these statistics.

Nimbra Vision comes with one Polling object, **DTM Node**. It creates the following statistics:

- **Interface Rx octets**. Collects the number of received (in) octets for all interfaces in the MIB-II ifTable
- **Interface Tx octets**. Collects number of transmitted (out) octets for all interfaces in the MIB-II ifTable.
- **Interface Rx discarded packets**. Collects the number of received and discarded packets for all interfaces in the MIB-II ifTable.
- **Interface Tx discarded packets**. Collects the number of discarded packets that should have been transmitted for all interfaces in the MIB-II ifTable.

© 2003-2010 Net Insight AB, All Rights Reserved

- **Trunk Tx (slots)**. Collects the used capacity in 512 kbps slots of all the DTM interfaces. The DTM interfaces represent the trunks. This collection is necessary for the on-the-fly reports that can be generated form the HTML client, and for the Reporter Reports, see Performance Reports.
- **Trunk Tx (Mbps)**. Collects the used capacity in Mbps for all the DTM interfaces. The DTM interfaces represent the trunks.

See Also

Monitoring Network Performance in the Web NMS User Guide

Performance Reports

Events (for G.826 performance reports)

Performance Reports

To display a collection of data collected by the performance module, select the statistics and choose menu **View | Plot | Collected Statistics** or **View | Plot | Current Statistics**.

You can set up a policy to generate daily, weekly and monthly reports for the DTM interface utilization. The reports contain summary and detailed information with graphs and tables for each DTM interface. To access these reports, choose the menu **View | Reporter Reports**. A navigation page will open in the web browser.

G.826 performance reports can be viewed in the Events database. The default configuration of Nimbra Vision also has Custom Views under the node **Performance** in the tree.

See Also

Generate Reports - policy for generating reports

Archived PM Reports

G.826 like performance reports are saved into a PM reports archive in the database. The purpose of the archive is to be able to keep a large number of performance monitoring reports over a long period of time. The function complements the **15 min** and **24 hr** views available below the **Performance** node in the tree. Those are actually custom views that selects PM reports from the events view. The limit of number of reports to store in the archive is set but the used database. As of MySQL 5.0, the is 2^{32} (~4.3 x 10^9) rows.

The following properties are included in the archived report:

- source
- objectName (Object)
- tag
- period (value `15m` or `24h`)
- modifiedInNode (Node Date/Time)
- time (Date/Time)
- IS (value `true` or `false`)
- ZS

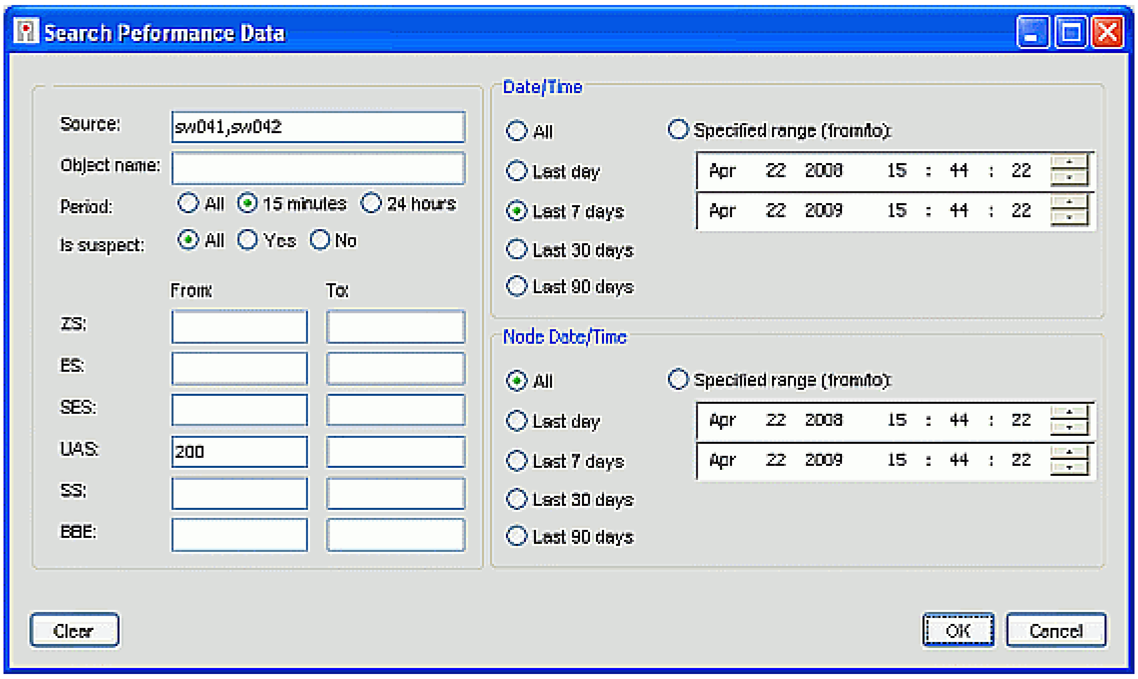
© 2003-2010 Net Insight AB, All Rights Reserved

- ES
- SES
- UAS
- BBE
- SS

See Events for description of the different properties.

Performance reports are added to the archive by an event filter. See Processing Events and Alarms.

To access the archived performance monitoring reports, open the **Archived PM Reports** node under the **Performance** node in the Tree. This opens a panel that contains a table of reports. To view some reports, use the menu **Edit | Search** (or use the toolbar button), which opens a search dialog. Enter the search criteria and click button **OK**. . The search criteria can use wildcards etc., see Writing Filer Criteria for description on how to write search criteria. The Archived PM Reports panel will be updated with events matching the entered criteria. The table is limited to 5000 rows.



Source: sw041,sw042
Object name:
Period: All [✓ 15 minutes] 24 hours
Is suspect: [✓ All] Yes No

	From	To
ZS:		
ES:		
SES:		
UAS:	200	
SS:		
BBE:		

Date/Time: All, Last day, [✓ Last 7 days], Last 30 days, Last 90 days, Specified range (from/to):
Apr 22 2008 15 : 44 : 22
Apr 22 2009 15 : 44 : 22

Node Date/Time: [✓ All], Last day, Last 7 days, Last 30 days, Last 90 days, Specified range (from/to):
Apr 22 2008 15 : 44 : 22
Apr 22 2009 15 : 44 : 22

Dialog for searching in the PM Reports Archive.

In the search dialog, you enter numerical ranges for the counters. If values are specified in both the **From** and the **To** columns for a counter, then counter values within that range are matched. If **From** column is empty, then all counter values less than the specified value in the **To** column is matched. If the **To** column is empty, then counter values greater than the value specified in the **From** column is matched.

A user is restricted to view only archived reports as specified by the Custom View Scope named **Events**.

See Also

Events

Cleanup Archived PM Reports

Processing Events and Alarms

Writing Filter Criteria

© 2003-2010 Net Insight AB, All Rights Reserved

About Services

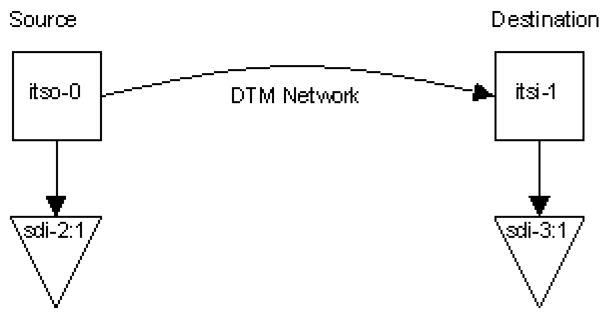
Introduction to Service Provisioning

Service Provisioning is the means of setting up or configuring services in the Nimbra network.

Nimbra Vision automatically discovers services when a node is being discovered. All services are added to the database as managed objects with classname *DtmService*. When a service is updated in the node, Nimbra Vision will automatically update the corresponding managed object in the database. This means that you can use Nimbra Vision together with the Element Manger, CLI or any other service provisioning system. Actually, when you modify a service from Nimbra Vision, the manage object is not updated until Nimbra Vision detects that the service is updated in the node.

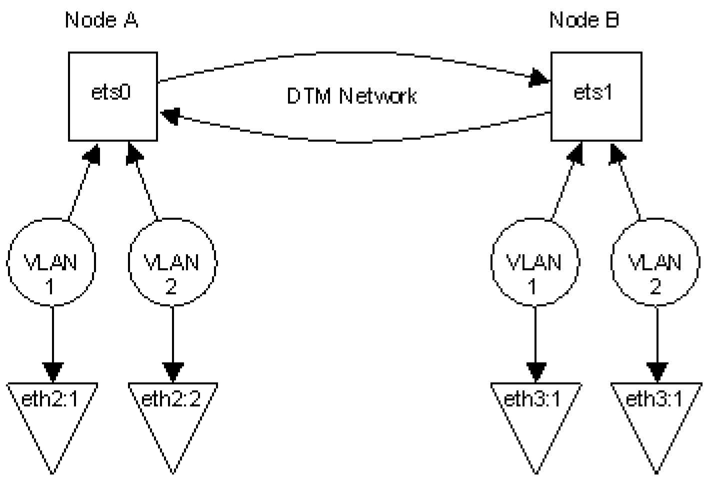
In the nodes, a service consists of a TTP (Trail Termination Point) in the originating node (source node), and a TTP in each terminating node (destination). A TTP is a logical entity that is used for originating and terminating connections (or channels), and for associating interfaces. Access interfaces are associated with the TTP's, either directly in the TTP (for ITS), or indirectly (for ETS). The originating TTP in the node also holds necessary data for the channels to the remote nodes, e.g. capacity, destinations etc. Nimbra Vision represents the service based on data available in the originating TTP.

***Note!** When Nimbra Vision sets up a service, it will use the hostname as known by Nimbra Vision as destinations; it will not use the DTM address. This makes it easier when using the Element Manager and CLI. For the node to be able to establish the connection it must have the same knowledge of the hostname, i.e. there must be an entry in the node's host list for the hostname to DTM address. You can use the HostListPush policy to ensure that the host list is synchronized with the Nimbra Vision database.*

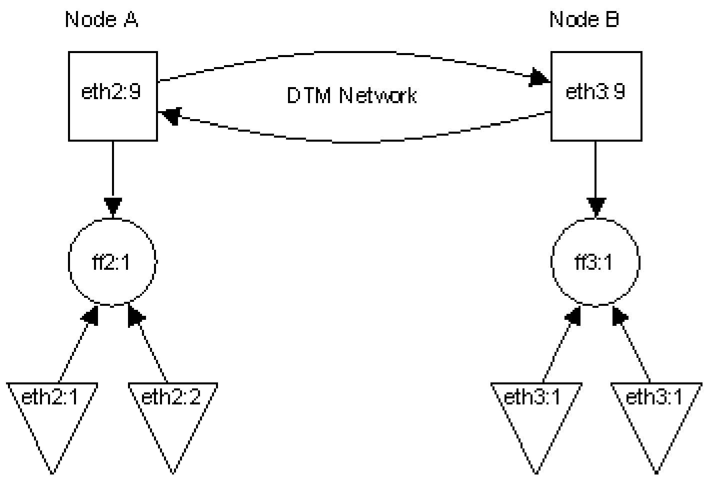


ITS service objects: The TTP in the originating node (itsi-0) has an association to the ingress access interface (sdi-2:1) and the TTP in the terminating node (itsi-1) has an association to the egress access interface (sdi-3:1). The originating TTP has associates the terminating TTP (or TTPs in case of multicast).

© 2003-2010 Net Insight AB, All Rights Reserved



ETS version 1 service objects: The TTPs (ets0, ets1) do not have any associations to the access interfaces. Instead, each interface is associated via a VLAN, and the VLAN is associated with the TTP.



ETS version 2 service objects: The TTP (eth2:9, eth3:9) has an association with the forwarding function (ff2:1, ff3:1). Other physical (eth2:1 etc.) and logical interfaces (none present) also have associations to the forwarding function. The TTP also has an association the TTP on the remote node.

Because Nimbra Vision has access to all nodes in a network, it automates and simplifies the process of managing services. When editing services, Nimbra Vision will be able to display data about both the originating and terminating node in the same window.

Some of the main operations related to services are:

- To add a service, see Adding a Service
- To edit an existing service, see Editing a Service
- To trace the channels, see Trace Channel
- To edit source routes, see List of Source Routes. You can also edit source routes from within the Service Editor.

Note! *Service provisioning requires a dedicated license key.*

Note! *It is essential that the nodes send* `config` *events to Nimbra Vision. These are sent as SNMP notification.*

See Also

Services Properties

© 2003-2010 Net Insight AB, All Rights Reserved

Source Routes

A source route is a description of a path through the network that can be used when establishing services. When configuring a service, it is possible to specify up to three source routes. When the service is being established, the node will attempt to use the first specified source route. If it fails to establish using this source route, it will try the second source route, and then the third. If no source route is specified, then the node will use a path with the lowest cost when establishing the service.

A source routes include an ordered specification of nodes, and optionally DTM interfaces.

A source route can be strict or loose:

- When a **strict** source route is used, then the channel must be established via all the nodes (and interfaces) in the specified order, and via no other nodes.
- When a **loose** source route is used, then the channel must be established via all the nodes (and interfaces) in the specified order, but the channel may be established via additional nodes as necessary.

A source route is a shared resource. Multiple services can use the same source routes.

A given source route is defined in one node, and can be used by services originating in that node only.

If the first source route is not used when a service is established, then an alarm can be raised.

See Also

Source Route Editor

Advanced Settings for Source for enable/suppression of alarm when not using first source route.

1+1 Service Protection

1+1 service protection is when a data stream is split and simultaneously sent in two channels through the network. The destination selects the currently "best" data stream. 1+1 service protection allows for very fast failover with minimal data loss. In a Nimbra network, support for 1+1 protection depends on the hardware and the type of service.

Normal 1+1 Service Protection

In normal 1+1 service protection, the protection is configured within a single service. The data stream is entering the network on a single ingress access interface, is split at the service source and transported in two channels to the service destination. The destination is selecting the "best" data stream to output at the egress access interface. The two channels, first channel and second channel, are configured on the service source, and one of these is selected at the service destination.

This type of protection can only be used on unicast service.

Open-ended 1+1 Service Protection

In open-ended 1+1 service protection, the protection is configured using two separate services that terminate on a shared service destination. The two services can originate on two different network elements, on two different ingress access interfaces on the same network element, or on the same ingress access interface. In the latter, the data stream is split internally. The shared service destination is

© 2003-2010 Net Insight AB, All Rights Reserved

selecting the "best" data stream to output at the egress access interface. The two channels are configured at each service, and one of these is selected at the destination.

This type of protection can be used on multicast services.

Selection of Active Channel

It is possible to manually switch the active channel when the destination is terminating two channels:

- **none**. No channel is established, no selection is available.
- **first**. The active channel is part of this service and is specified as **First channel** on the Source side in the Service Editor.
- **second**. The service is 1+1 protected. The active channel is part of this service and is specified as **Second channel** on the Source side in the Service Editor.
- **this**. The service is protected using open-ended 1+1. The active channel is originating on the service displayed in the Service Editor.
- **other (*serviceName*)**. The service is protected using open-ended 1+1. The active channel is originating in the other service that shares the destination, named *serviceName*.

Service Provisioning

Adding a Service

To add a service, select a node in a map or in the network database, and select the menu **Nimbra Node | Add Service…**. This will open a wizard that helps you create a new service.

If you select multiple nodes, then the remaining nodes are considered to be destinations, i.e. where the service terminates.

The service will originate in the first selected node.

The wizard asks for the following information:

- **Customer ID** is a number that can be used to identify a customer using the service.
- **Source node** is the node where the service shall originate. This combo-box contains all nodes that were selected when the wizard was started. Select one of the nodes, type the correct node, or find another node by opening the search dialog using the button **Add node…**.
- **Service type** is one of *ITS unicast*, *ITS multicast*, *ETS unicast* or *ETS multicast*.
- **Source interface**. (Only for ITS). This is the ingress interface. If an input interface is in use by another service (i.e. if its operational state is up), then this is indicated with `[busy]` after the interface name. If the interface is absent then this is indicated with `[absent]`. The node reports an interface as absent if the hardware is missing, or if multiple interfaces (e.g. an ASI and an SDI) share the same physical port and the port is in use by another interface (which is indicated by `[busy]` on that other interface).
- **Source device**. (Only for ETSv2). This is the device where the ETS service will originate.

© 2003-2010 Net Insight AB, All Rights Reserved

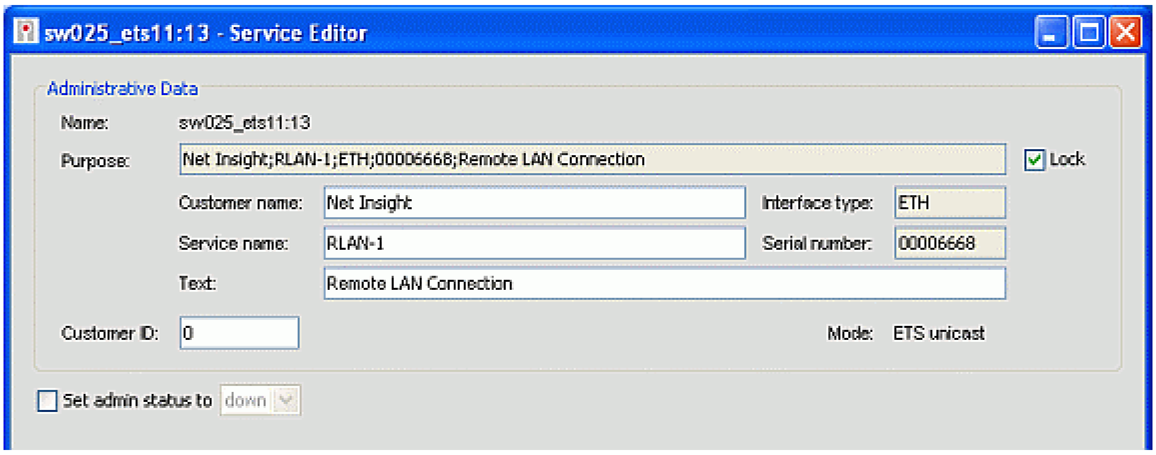
- **Destination interface**. (Only for ITS). This is the ingress interface (or interfaces in case of multicast).
- **Destination device**. (Only for ETSv2) This is the device (or devices is multicast) where the service will terminate.

When the wizard is finished, it will create the service in the nodes based on the entered data. If multiple nodes were selected when the wizard was started, then destinations will be created for all these nodes (except for the one used as source). It will then start the applicable Service Editor. From here you can continue setting up the service.

Editing a Service

To edit a service, open the Service Editor for the service. To open the service editor, select the service in the network database, and select menu **Service | Edit…**. This opens a service editor dedicated for the selected type of service. The service editor is also opened when the Add service wizard is finished and has created the service.

When the Service Editor is opened, it reads information from the originating and all terminating nodes.


sw025_ets11:13 - Service Editor

Administrative Data

Name: sw025_ets11:13

Purpose: Net Insight;RLAN-1;ETH;00006668;Remote LAN Connection ☑ Lock

Customer name: Net Insight | Interface type: ETH

Service name: RLAN-1 | Serial number: 00006668

Text: Remote LAN Connection

Customer ID: 0 | Mode: ETS unicast

☐ Set admin status to down

The top part of all Service Editor windows contains the same type data: Administrative data, and Global admin status control.

The Service Editor are slightly different for ITS and ETS, and for unicast or multicast services. The top part of the Service Editor always contains the following information.

The **Administrative data** section contains data of administrative nature. Its purpose is only to aid in identifying the service. None of the data is required for operation of the service. The data can be modified at any time without affecting the operation of the service.

- **Name** is the name of the managed object in the network database. Note that this is not the same as **Service Name** (see below).
- **Purpose** is a free text describing the service. This information is also available on alarms and alarm events from the nodes. The purpose text is available in the node, and can be seen using the Element manger or CLI. Nimbra Vision can manage the Purpose as a concatenation of the sub-fields Customer name, Service name, Interface type, Serial number, and Text fields, which are separated with semi-colons (";"). If Nimbra Vision is able to detect the sub-fields from the purpose string, then it will present them as the separated fields. Otherwise, it will present the purpose as a single string. You can switch back and forth between the two formats using the **Lock** checkbox. To be able to directly edit the

© 2003-2010 Net Insight AB, All Rights Reserved

Purpose, you must un-tick the **Lock** checkbox. Whenever it is ticked, then the Purpose is updated based on the sub-fields. The maximum size for the Purpose is 255 characters.

- **Customer name**. This is a text field, typically used for specifying the name of the customer or user using the service. This is a sub-field to the Purpose field.
- **Service name** is a text field where you can name the service. Note that this is not the same as the name on the service managed object (*DtmService*) in the network database. This is a sub-field to the Purpose field.
- **Interface type** is derived form the interface. This is ETS for ETS service, and otherwise the first part of the interface name (e.g. ASI, SDI, PDH etc.). This is automatically assigned when the service is created. This is a sub-field to the Purpose field.
- **Serial number** is a serial number assigned to the service when Nimbra Vision creates it. The serial number is incremented by one for each created service. An administrator can set the initial serial number; see Setting of Initial Serial Number for Services. This is a sub-field to the Purpose field.
- **Text** is free text. This is a sub-field to the Purpose field.
- **Customer ID** is a non-negative integer that can be used for identifying a customer. This data exists in the node, and is a complement to the Purpose in the node, or the Customer name.
- **Mode** describes the mode of the service, and also describes the type of Service Editor; *ETS unicast*, *ETS multicast*, *ITS unicast*, or *ITS multicast*.

Additional data common for all Service Editor types are:

- **Set admin status to**. Using this checkbox you will affect the administrative state (**Admin**) on all objects related to this service on all nodes. This is a quick way to affect both the originating TTP, and all terminating TTP's, and it is the normal procedure to enable or disable the service. When ticking the checkbox, you can select to set the admin status to *up* or *down*.

The part of the Service Editor below the common section is the section specific to the type of service. Typically, it contains a source column and a destination column representing settings on the source and all the destinations (multiple in case of a multicast service). See the chapters describing each type of service for details.

The button **Trace Channels…** near the bottom of the window, opens up the trace channel dialog for the channel related to the service.

At the bottom of the window, the following buttons affect the service as:

- **Refresh** discards all edited changes and reads all the data anew from the nodes.
- **OK** commits the edited changes and closes the window.
- **Apply** commits the edited changes and refreshes the window.
- **Cancel** discards all unapplied changes and closes the window.
- **Delete** opens a window for you to confirm deletion of the service or part thereof.

At the far bottom of the window is the *status bar*. It shows messages of the status of the service as presented by the node such as error messages. It also shows

© 2003-2010 Net Insight AB, All Rights Reserved

work in progress. If the status bar is clicked when it is showing a message, the message will displayed in a new window.

See Also

ITS Service Editor (unicast)

ITS Service Editor (multicast)

ETS Service Editor (unicast)

ETS Service Editor (multicast)

Setting of Initial Serial Number for Services.

Advanced Settings for Source

This dialog is opened from the Service Editor. None of the settings in this dialog is applied until the data in the Service Editor is applied.

- Setting of **DSTI**. The DSTI is a number that uniquely identifies the TTP within the node. Nimbra Vision normally automatically assigns this, but you can also set or change the DSTI manually.
- **Connection Re-establishment** interval parameters exponential back-off algorithm for re-establishing channels:
    - **Minimum interval**. The start value of the algorithm. After a teardown of the connection, it will try immediately to re-establish the connection, if it fails it will wait this number of milliseconds and then retry.
    - **Maximum interval**. The end value of the algorithm. The reestablish mechanism will wait no longer than this number of milliseconds.
- **Precedence**. This is a function that gives the operator the possibility to give a channel precedence to be torn down and re-established. This is especially valuable at fault situations where this gives channels with precedence a chance to be torn down and re-established first. This function will reduce the set-up time for critical channels and as far as possible allocate the free capacity in the network to these channels. There are two levels of precedence, either the channel has precedence or not. Precedence should only be setup on a small number of channels per node, recommended is no more than five.
- **Suppress first source route alarm**. If a channel to the destination is not setup using its first source route, then an alarm can be raised to indicate this. This setting controls suppression of this alarm.

Advances Settings for Destination

This dialog is opened from the Service Editor. None of the settings in this dialog is applied until the data in the Service Editor is applied.

- **Set DSI as**. The destination TTP is per default automatically assigned a DSTI when Nimbra Vision creates the TTP. This dialog allows the operator to change the DSTI, or to associate the service with another already existing TTP. In some cases, you want to establish a connection to an already existing TTP. One such common situation is when you want to establish a bidirectional multicast ETS connection. In this case, the return connection should be configured to be established to the

© 2003-2010 Net Insight AB, All Rights Reserved

already existing TTP. Use this dialog to configure to use the already existing destination.

- **Delete the current destination**. Per default, when a destination is changed or deleted in the Service Editor, then the corresponding terminating TTP is also deleted. This checkbox allows the operator to keep the terminating TTP. This can be used when one of the services in an open-ended 1+1 configuration shall be deleted, but the other service shall be kept.

Deleting a Service

When a service is deleted using the Delete button in the Service Editor, then the service is deleted on the source node and on all the destination nodes.

Optionally, you can choose to not delete the data on the terminating nodes. This option is useful only if you know that you will use the terminating TTP for some other service, e.g. if the destination is shared using open-ended 1+1 service protection.

If the service is an ETSv1 service, then the associated VLANs on originating and terminating nodes are also deleted.

No interfaces or forwarding functions are deleted when the service is deleted.

Services Properties

Services are represented as Managed Objects, just as nodes, interfaces and links.

Property	Description
administrativeState	The administrative state of the service. This is retrieved from the TTP in the originating node. The administrative state is the desired state of the service, *up* or *down*. When *up*, the service is enabled, and when *down*, the service is disabled. Also, see **operationalState**.
chmgrODescrIndex	This integer value is the index in the `chmgrODescrTable` in the NETI-CHMGR-MIB representing this originating service. It is used internally.
classname	The classname is always *DtmService* for managed objects representing services.
name	The name of the service. The name is a string based on the name of the originating node and the name of the originating TTP as defined by the node. This is the key in the database. This is not the same as the Service name defined as a part of the Purpose in the Service Editor, and stored in **ttpPurpose**.
operationalState	The operational state is the actual status of the service. This is retrieved from the TTP in the originating node. The following operational states exist: *up* - the service is up and fully functional to all destinations. *lowerLayerDown* - a fault is detected on the

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
	originating access interface (**origAccessInterface**). *partial* - the multicast service is up to at least one, but not to all destinations, or only one of the channels in a 1+1 protected service is up. *down* - the service is not up to any of its destinations.
origAccessInterface	The name of the access interface where the service originates, as known by the node. For ITS (Video/PDH/SDH etc.) services, this is the name of the access interace. For ETSv2 (Ethernet), this is the name of the forwarding function. For ETSv1, interface is associated using VLAN's, and the string *n/a* is presented.
origDtmNode	The name of the node where the service originates.
requestedCapacity	The requested capacity in bits per second (bps) for the service. This value is always 0 if the capacity cannot be set on the service.
serialNumber	When Nimbra Vision creates a service, it will assign the service a serial number. This serial number is a sequence number starting with 1 and increased by one for each new service created. The serial number will also be a part of the purpose, see **ttpPurpose**. See Setting of Initial Serial Number for Services.
state	The updated state of the object. Describes how the data in the managed object is or needs to be updated with the data in the node: *OK* - The managed object is up to date. Its data reflects the data in the node. *Not updated* - An event has been received indicating that data in the node has changed and that the managed object must be updated to reflect this change. The managed object is queued to read the data from the node to get updated. *Update failed* - Indicates failure when attempting to read data from the node when updating the managed object. This state would be set if e.g. there is a communication error while updating the managed object. Will try again.
status	The severity of the service is derived from the administrative and operational state (**administrativeState** and **operationalState**) as: *down*/*down* - clear *up*/*up* - clear *up/lowerLayerDown* - minor *up*/*partial* - major *up*/*down* - critical Any other combination - major. Note that the node generates an alarm when a service has a fault. This alarm is tracked on the node in Nimbra Vision, and will therefore affect the status on the node managed object and not the service managed object. But, faults will also affect the operational status (**operationalState**), and hence indirectly affect the status on the service managed object.

© 2003-2010 Net Insight AB, All Rights Reserved

Property	Description
tag	A user defined string that can be used for tagging the managed object. See Tagging Managed Objects.
termAccessInterface	The name of the access interface where the service terminates, as known by the node. This is only presented for unicast. For ETSv2, this is the name of the forwarding function. For multicast, the string *multiple* is presented instead. For ETSv1, the interface is associated using VLAN's, and the string *n/a* is presented.
termDtmNode	The name of the terminating node. If the service is multicast, then the string *Multiple* is presented instead. Note that node names always use lower case letters, so the string *Multiple* will never match the name of an existing node.
ttpCustomerId	A customer identity. This is an integer value. This property is an administrative value that can be used to identify different customers for different services. It is assigned by the user or operator.
ttpDestinationDsti	The DSTI (DTM Service Type Instance) of the TTP on the terminating node for unicast services. For multicast services, this value is always 0.
ttpIndex	The index in the SNMP table representing the service. The service is represented by an entry in a SNMP table on the node where the service originates. The index represents the row in that table. If the service type is ITS, then the index represents a row in the table `itsSrcTtpTable` in the NETI-ITS-MIB. If the service type is ETS, then the index represents a row in the table `dltTtpTable` in the NETI-DLT-MIB, or in the tables `ethIfTable` and `ethEtsTable` in the NETI-ETH-MIB. This is used internally.
ttpLocalDsti	The DSTI (DTM Service Type Instance) of the TTP on the originating node.
ttpMode	The value is *unicast* or *multicast*.
ttpPurpose	The purpose of the service. The purpose is an administrative string assigned by the user or operator. The Purpose property is present in many locations where it is of use to identify a specific service; in alarm events, in alarms, when setting up Redundant Headends, etc.
ttpServiceType	The type of service, as further refined from **type**. The type is based on the name of the access interface. Some types are: *ets* *dvb* *sdi* *sdh* *pdh*
type	The type of managed object this is: *ETS* - Represents an Ethernet Transport Service. *ITS* - Represents an Isochronous Transport Service. This includes transport of video (HD-SDI, SDI and ASI), audio (AES/EBU), PDH and SDH.

© 2003-2010 Net Insight AB, All Rights Reserved

List of Source Routes

The Source Routes dialog shows all source routes defined in the node. To access this dialog, select the node for which the source rotes are defined, either in a map or in the network database, and select menu
Nimbra Node | Edit Source Routes….

A list of source routes is retrieved from the node when the dialog is opened. The dialog is not updated if source routes are added or deleted in the node while the dialog is open.

- Use button **Used By…** to open a dialog that lists all services using the selected source route. From here you can open the Service Editor to edit the service that is using the source route.
- Use button **Add…** to add a new source route to the node. This opens the Source Route Editor.
- Use button **Edit…** to edit the selected source route. This opens the Source Route Editor for the selected source route.
- Use button **Delete** to delete the selected source route from the node. A source route that is used by a service cannot be deleted.

See Also

Source Route Editor

Editing a Service

Source Routes

Source Route Editor

The Source Route editor is opened from a selected source route in the List of Source Routes window. The editor allows editing of a single source route

A source route can be reused by any service that originates in the node.

- **Source route name**. Enter a name to identify the source route.
- **Outgoing interface**. The name of the DTM interface used when leaving the node. This is optional.
- **Routing**. Radio button defining if routing is **strict** or **loose**.
- **Route**. A list of hops defining the route. Interface is optional.

Buttons to the right affects the list:

- **Add** a new row in the route list above the selected row.
- **Delete** the selected row.
- **Add Highlighted**. Adds all nodes and interfaces (represented by links) that are highlighted in the maps.
- **Move Up**. Moves the selected row up in the table.
- **Move Down**. Moves the selected row down in the table.

The buttons at the bottom applies to the defined source route:

- **Verify**. Checks the source route for some errors. To pass, all defined nodes must be known, and in case of a strict source route, all hops must be to a neighboring node.
- **Highlight** the defined source route in the maps.
- **OK** applies the changes and closes the window.

© 2003-2010 Net Insight AB, All Rights Reserved

- **Apply** applies the changes.
- **Cancel** closes the window without applying any changes.

> ***Tip!*** *A quick way to add a source route is to select nodes (and optionally interfaces) in the maps, highlight the selected objects, and then add the highlighted to the editor with the button* ***Add Highlighted****.*

See Also

Source Routes

Select Node

The select node dialog searches the database according to search criteria. One of the results can be selected.

To search for the node, enter search criteria and click the button **Search**.

- **Node name**. Only nodes matching this criterion will be displayed.
- **Node type**. Only nodes matching this criterion will be displayed.

Results form the search is displayed in the Search results table. For each node, the name, type and location is displayed.

To select a node, select the row in the table corresponding to the node, and click button **Select**.

See Also

Writing Filter Criteria

Operations on Services

You can perform different operations on the managed objects representing the service. To perform an operation, select the service in the network database. This will enable the object menu **Nimbra Service**. You can also use the right mouse button on the object to open the menu as a pop-up menu.

The available operations depend on your permissions, as defined by the administrator.

Operations

- Edit.... Opens the selected service in the Service Editor.
- Add.... Opens the wizard to add a new service that is originating on the same node as the selected service.
- **Enable at Source**. Sets the administrative state to *up* on the originating side of the service (i.e. on the originating trail termination point, TTP).
- **Disable at Source**. Sets the administrative state *down* on the originating side of the service (i.e. on the originating trail termination point, TTP).
- Trace Channel. Start a trace of the channel represented by the selected service through the network.
- Managed Object Properties. Opens a dialog to display and edit the data about the service in the Nimbra Vision database.
- **Delete from Database**. Deletes the data about the service from the Nimbra Vision database. No data is affected in the nodes.

© 2003-2010 Net Insight AB, All Rights Reserved

SNMP Direct

The functions SNMP Direct allows you to set how the client shall access the network elements when performing different operations like Service Provisioning, List Channels, Trace Channel etc.

When the node is configured as a result of user functions in the client, command must be sent to the node to actually do the configuration. Nimbra Vision is using SNMP for this. Typically, this would be when a service is created or changed.

Nimbra Vision client has two modes of operation for this:

- **via the server**. In this mode, the client communicates with the server. The server sends the SNMP request to the node and receives the response. The response is the communicated back to the client. The advantage with this solution is that the client is never directly accessing the nodes, which means that the client can run outside a firewall.
- **directly from the client**. In this mode, the client communicates directly with the nodes. It sends the SNMP request to the node, and receives the response. The advantage with this mode is that it is faster. This mode requires the client to have direct SNMP access to the nodes.

To set the SNMP direct mode for the current session, choose menu **Tools | Set SNMP Direct…** and select desired mode. This will take immediate effect.

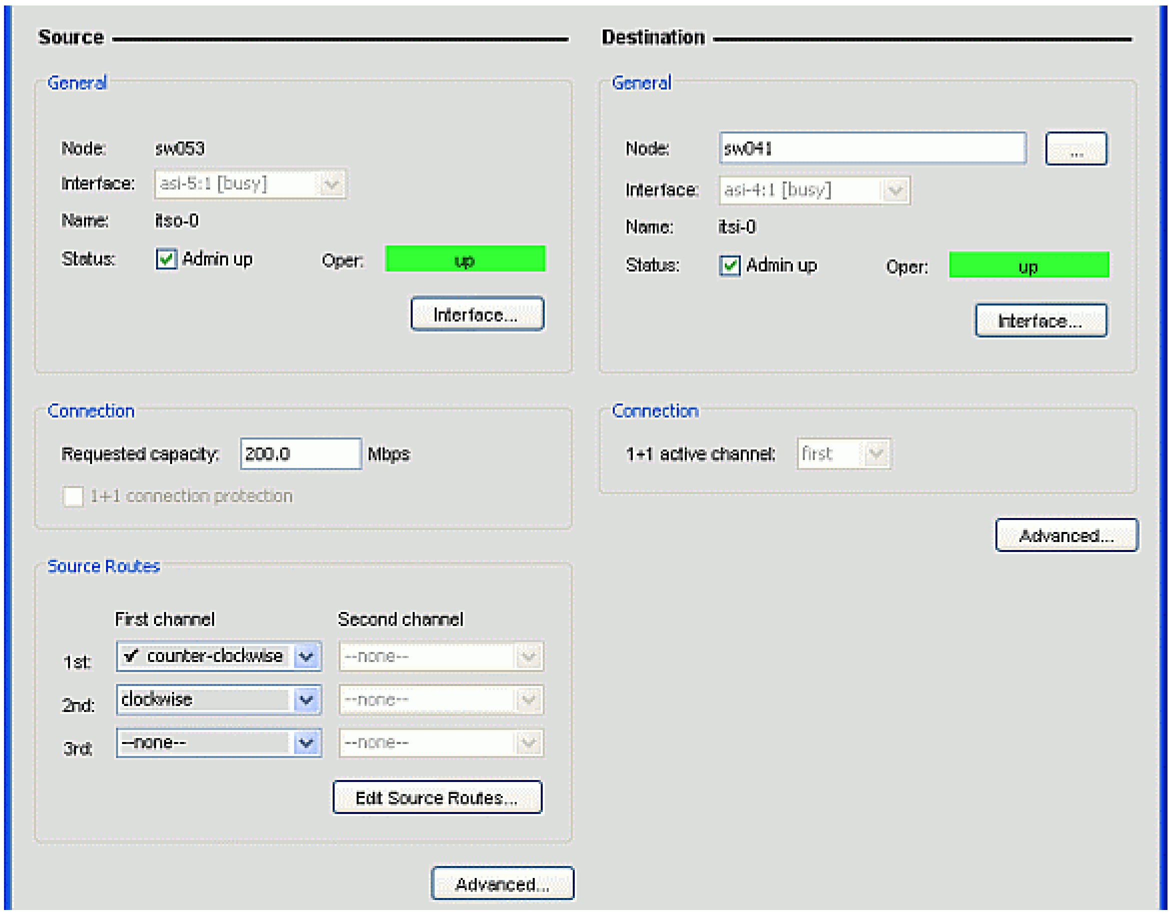
To permanently configure the default value, set the parameter `SNMP_DIRECT` in the configuration file *<NMS_Home>*`\conf\clientparameters.conf` to `true` or `false`.

Isochronous Transport Services, ITS

ITS Service Editor (unicast)

The top part of the Service Editor contains the **Administrative data** and the Global admin status control. This section is the same on all variants of the service editor. See section Editing a Service for the general description of this part.

© 2003-2010 Net Insight AB, All Rights Reserved


Source

General

Node: sw053

Interface: asi-5:1 [busy]

Name: itso-0

Status: ✓ Admin up Oper: up

Connection

Requested capacity: 200.0 Mbps

1+1 connection protection

Source Routes

	First channel	Second channel
1st:	✓ counter-clockwise	--none--
2nd:	clockwise	--none--
3rd:	--none--	--none--

Destination

General

Node: sw041

Interface: asi-4:1 [busy]

Name: itsi-0

Status: ✓ Admin up Oper: up

Connection

1+1 active channel: first

The connection specific configuration of an ITS unicast service.

Source

The left half of the Service Editor under the heading **Source** represents the Source data. This data is representing the TTP on the originating node.

The **General** section:

- **Node** is the name of the node where the service originates. This is defined when the service is created, and cannot be changed.
- **Interface** is a combo-box listing all interfaces on the node for the type of service. The administrative state of the service must be down to be able to change interface. An interface where then operational state is up is marked with `[busy]`.
- **Name** is the name of the TTP in the originating node. The name is automatically generated.
- **Admin up** controls the administrative state of the originating TTP. The normal way to control the administrative state of the service is to control all TTPs in a single operation using the **Set admin status to control** at the general part of the page, see chapter Editing a Service.
- **Oper** displays the operational state of the TTP.
- **Interface…**. This button opens a dialog for the selected interface. See chapter ITS Service Interface.

The **Connection** section:

- **Requested capacity**. If the service supports setting of the capacity, then this is specified here.

© 2003-2010 Net Insight AB, All Rights Reserved

- **1+1 connection protection**. If the service supports use of 1+1 protection using two channels from the same TTP, then its use is specified here. When a service is 1+1 protected, the data is split in the originating interface and sent in two channels to the destination. The destination selects to listen to one of the channels. Check this checkbox if 1+1 protection shall be used. If 1+1 protection is used, then the two channels must be source routed.

The **Source Routes** section contains a matrix for assigning source routes to the connection. A source route is a definition of a path through the network. If a service is 1+1 protected, then the two channels must have at least one source route each, and the source routes should define different paths through then network as much as possible. The information is displayed in a 2 × 3 matrix as:

- **First channel/Second channel**. The columns represent the first and the second channel in case of a 1+1 protected connection. If 1+1 protection is not used, then there is only a first channel.
- **1st/2nd/3rd**: The rows represent what source route to use as first, second or third choice. If a source route is not specified, it is presented as `--none--`. The name of the currently used source route is marked with a tick. See chapter Source Routes for details.
- **Edit Source Routes…**. This button accesses dialogs for editing and displaying source routes on the node where the service originates. See chapter List of Source Routes.

The **Advanced…** button at the end of the Source part of the page opens a dialog for advanced settings, see chapter Advanced Settings for Source.

Destination

The right half of the Service Editor under the heading **Destination** represents the Destination data. This data represents the terminating TTP. If it is a multicast service, then all destinations are listed in a table.

The **General** section:

- **Node** is the name of the node where the service terminates. Changing this will modify the destination defined in the source TTP, and will add/delete a terminating TTP in the destination nodes.
- **Interface** is a combo-box listing all interfaces on the node for the type of service. An interface where then operational state is up is marked with `[busy]`. On some interface boards, the interface implemented as two physical ports, one for input and one for output. On these boards, the operational state is reflecting the originating port.
- **Name** is the name of the TTP in the terminating node. The name is automatically generated.
- **Admin up** controls the administrative state of the terminating TTP. The normal way to control the administrative state of the service is to control all TTPs in a single operation using the **Set admin status to control** at the general part of the page, see chapter Editing a Service.
- **Oper** displays the operational state of the TTP.
- **Interface…**. This button opens a dialog for the selected interface. See chapter ITS Service Interface.

The **Connections** section:

- **1+1 active channel**. If the service is 1+1 protected, then this combo-box displays the active channel, as **none**, **first**, **second**, **this** or **other**. It is possible to manually switch the active channel. See chapter 1+1 Service Protection.

© 2003-2010 Net Insight AB, All Rights Reserved

The **Advanced…** button at the end of the Destination part of the page opens a dialog for advanced settings, see chapter Advanced Settings for Destination.

See Also

Editing a Service for details common to all Service Editors.

ITS Service Interface for configuration of the service interface.

List of Source Routes for details on configuring and displaying source routes.

Source Routes for general description of source routes.

Advanced Settings for Source for details on how to configure advanced settings such as DSTI, precedence etc.

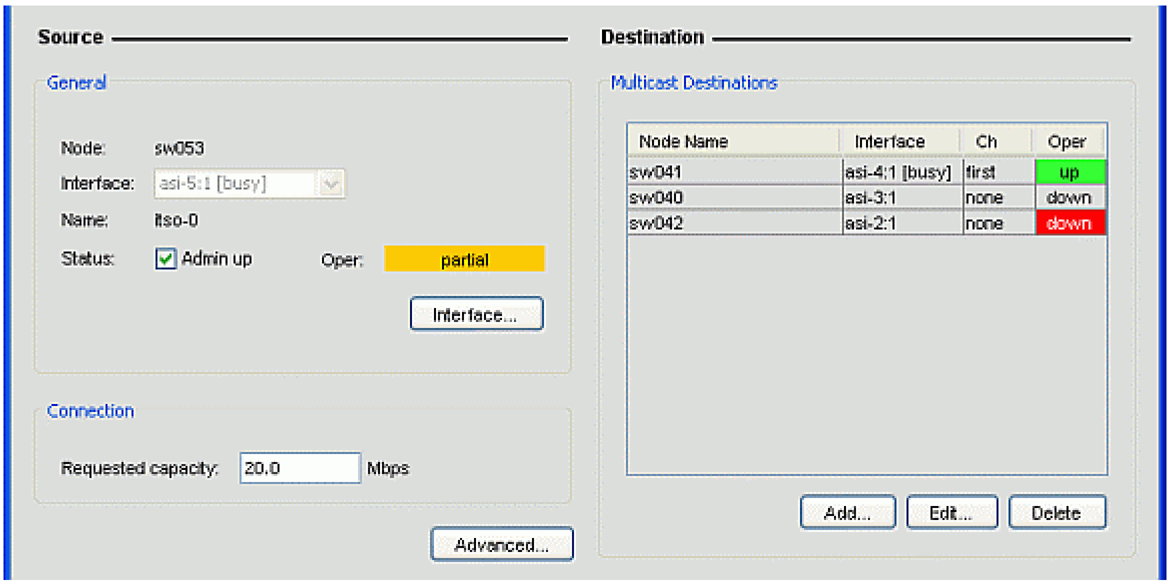
1+1 Service Protection for general description of 1+1 service protection.

Advanced Settings for Destination for details on how to configure advances settings such as DSTI.

ITS Multicast Destination

ITS Service Editor (multicast)

The top part of the Service Editor contains the **Administrative data** and the Global admin status control. This section is the same on all variants of the service editor. See section Editing a Service for the general description of this part.


Source

General

Node: sw053

Interface: asi-5:1 [busy]

Name: itso-0

Status: ✓ Admin up   Oper: partial

Connection

Requested capacity: 20.0 Mbps

Destination

Multicast Destinations

Node Name	Interface	Ch	Oper
sw041	asi-4:1 [busy]	first	up
sw040	asi-3:1	none	down
sw042	asi-2:1	none	down

The connection specific configuration of an ITS multicast service.

Source

The left half of the Service Editor under the heading **Source** represents the Source data. This data is representing the TTP on the originating node.

The **General** section:

- **Node** is the name of the node where the service originates. This is defined when the service is created, and cannot be changed.
- **Interface** is a combo-box listing all interfaces on the node for the type of service. The administrative state of the service must be down to be able to change interface. An interface where then operational state is up is marked with `[busy]`.
- **Name** is the name of the TTP in the originating node. The name is automatically generated.

© 2003-2010 Net Insight AB, All Rights Reserved

- **Admin up** controls the administrative state of the originating TTP. The normal way to control the administrative state of the service is to control all TTPs in a single operation using the **Set admin status to control** at the general part of the page, see chapter Editing a Service.
- **Oper** displays the operational state of the TTP.
- **Interface…**. This button opens a dialog for the selected interface. See chapter ITS Service Interface.

The **Connection** section:

- **Requested capacity**. If the service supports setting of the capacity, then this is specified here.

The **Advanced…** button at the end of the Source part of the page opens a dialog for advanced settings, see chapter Advanced Settings for Source.

Destination

The **Multicast Destinations** section displays the multicast destinations. The section contains a table where each destination is represented by a row. The table displays the node, the active channel (if using open-ended 1+1 service protection), the operational status to the destination, and the service interface. Below the table are the following buttons:

- **Add**. Opens a dialog for adding a destination. See chapter ITS Multicast Destination.
- **Edit**. Opens a dialog for editing the destination selected in the table. See chapter ITS Multicast Destination.
- **Delete**. Deletes the destination selected in the table.

See Also

Editing a Service for details common to all Service Editors.

ITS Service Interface for configuration of the service interface.

List of Source Routes for details on configuring and displaying source routes.

1+1 Service Protection for general description of 1+1 service protection.

Advanced Settings for Source for details on how to configure advanced settings such as DSTI, precedence etc.

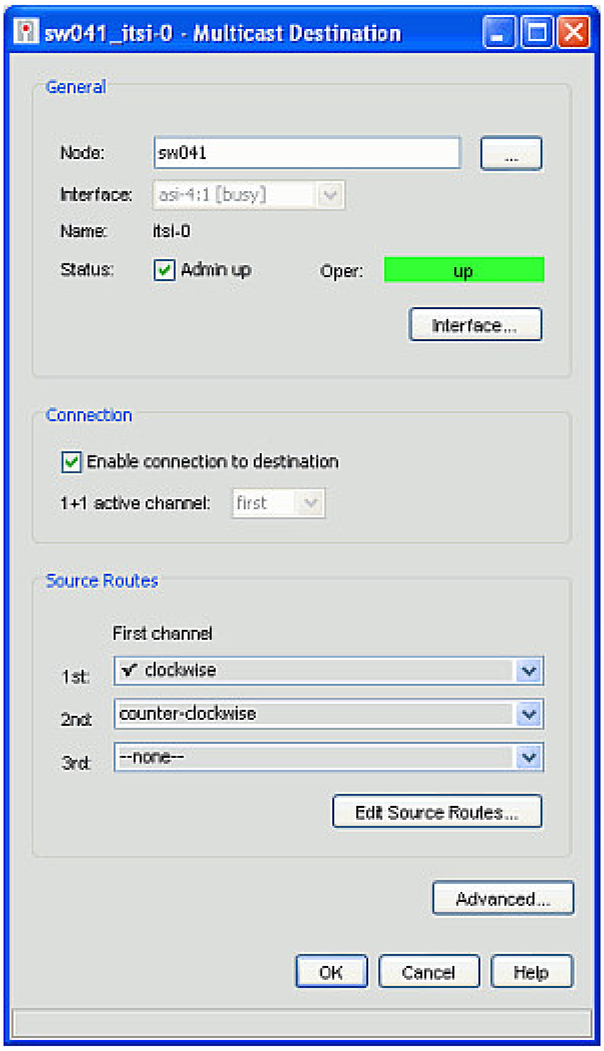
Advanced Settings for Destination for details on how to configure advances settings such as DSTI.

ITS Multicast Destination

ITS Multicast Destination

This dialog describes a single destination for an ITS multicast service. It is similar to the **Destination** section in the Service Editor for a unicast service. The dialog is opened from the Service Editor.

© 2003-2010 Net Insight AB, All Rights Reserved


sw041_itsi-0 - Multicast Destination

General

Node: sw041

Interface: asi-4:1 [busy]

Name: itsi-0

Status: ✓ Admin up Oper: up

Connection

✓ Enable connection to destination

1+1 active channel: first

Source Routes

First channel

1st: ✓ clockwise

2nd: counter-clockwise

3rd: --none--

ITS multicast destination dialog.

The **General** section:

- **Node** is the name of the node where the service terminates. Changing this will modify the destination defined in the source TTP, and will add/delete a terminating TTP in the destination nodes.
- **Interface** is a combo-box listing all interfaces on the node for the type of service. An interface where then operational state is up is marked with `[busy]`. On some interface boards, the interface implemented as two physical ports, one for input and one for output. On these boards, the operational state is reflecting the originating port.
- **Name** is the name of the TTP in the terminating node. The name is automatically generated.
- **Admin up** controls the administrative state of the terminating TTP. If this is unchecked, then the terminating TTP will be down and will not accept termination of the connection. Note that the normal way to control the administrative state of the service is to control all TTPs in a single operation using the **Set admin status to control** at the general part of the page, see chapter Editing a Service.
- **Oper** displays the operational state of the TTP.
- **Interface…**. This button opens a dialog for the selected interface. See chapter ITS Service Interface.

© 2003-2010 Net Insight AB, All Rights Reserved

The **Connection** section:

- **1+1 active channel**. If the service is 1+1 protected, then this combo-box displays the active channel, as **none**, **first**, **second**, **this** or **other**. It is possible to manually switch the active channel. See chapter 1+1 Service Protection.
- **Enable connection to destination**. This controls whether the connection shall be established to the configured destination. You can use this to disable the connection to a single destination without removing the configuration of the destination.

The **Source Routes** allows for assigning source routes to the connection. A source route is a definition of a path through the network:

- **1st/2nd/3rd**: The rows represent what source route to use as first, second or third choice. If a source route is not specified, it is presented as `--none--`. The name of the currently used source route is marked with a tick. See chapter Source Routes for details.
- **Edit Source Routes…**. This button accesses dialogs for editing and displaying source routes on the node where the service originates. See chapter List of Source Routes.

The **Advanced…** button opens a dialog for advanced settings, e.g. setting of DSTI etc., see chapter Advanced Settings for Destination.

See Also

Editing a Service

ITS Service Editor

ITS Service Interface

List of Source Routes for details on configuring and displaying source routes.

Source Routes for general description of source routes.

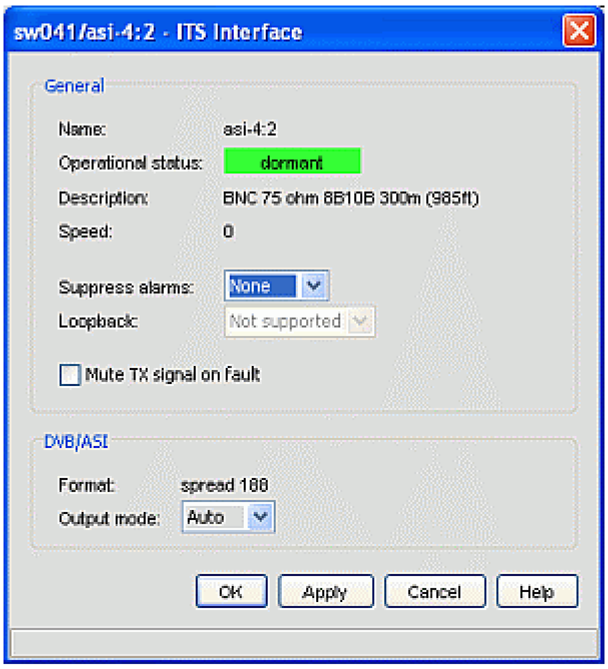
1+1 Service Protection for general description of 1+1 service protection.

ITS Service Interface

This dialog is opened from the Service Editor. Applying changes will be set immediately.

In this dialog, you can change settings for the access interface. The exact details may vary on different types of access interfaces.

© 2003-2010 Net Insight AB, All Rights Reserved



sw041/asi-4:2 - ITS Interface

General

Name: asi-4:2

Operational status: dormant

Description: BNC 75 ohm 8B10B 300m (985ft)

Speed: 0

Suppress alarms: None

Loopback: Not supported

☐ Mute TX signal on fault

DVB/ASI

Format: spread 188

Output mode: Auto

DVB/ASI interface as an example of an ITS access interface.

General

These settings apply for all type of access interfaces.

- **Name**. Then name of the interface as given by the node. The name normally consists of a
- **Operational status**:
    - **up**, the interface is in use be an originating service, i.e. the ingress interface is in active use.
    - **down**, the interface is not in use by any service
    - **dormant**, the interface is ready to be used by a service, but no service is originating in the interface.
    - **absent**, the physical hardware (e.g. the board) having the interface is not present in the node, or multiple interfaces (e.g. ASI and ASI) share the same physical port, and the port is used by one of the other interfaces.
- **Description**
- **Speed** interface line speed
- **Suppress alarms**. Faults detected on the access interface can normally generate alarms. Use this to suppress some or all alarms.
- **Loopback**. For testing and diagnostics purpose, you can loop back the data received on the interface.
    - **None**, this is the normal operation on the interface.
    - **Line**, data received in the physical interface is looped back to the physical interface.
    - **DTM**, data received from the far interface from the network is looped back to the far interface. This tests the Nimbra network.
- **Mute TX signal on fault**. If checked, the transmit port is muted upon reception of a defect indication such as AIS on the terminating TTP. If not checked, the port transmits IDLE bytes (ASI) or Undefined (other boards) during defect intervals. The muting function may speed up fail-

© 2003-2010 Net Insight AB, All Rights Reserved

over switching when using external fail-over switches.

PDH

These settings apply only for PDH type of interfaces:

- **Signal to transport**:
    - **E3**.
    - **DS3**.
- **Framing**:
    - **G.751**. Only when transporting E3 signal.
    - **G.832**. Only when transporting E3 signal.
    - **C-bit**. Only when transporting DS3 signal.
    - **M13**. Only when transporting DS3 signal.

SDH/Sonet Section

- **Interface mode**.
    - **SDH**.
    - **Sonet**.
- **Transmit sync source**. This parameter determines which clock is used for the outgoing signal from the interface:
    - **Loop**. The clock from the incoming signal on the interface is extracted and used as clock or the outgoing signal.
    - **Internal**. The node internal clock is used as clock for the outgoing signal.
- **Sync status message (SSM)**. Select value in range 1-15. Sets the outgoing SSM in byte S1.
- **Section overhead bytes (SOH)**. A set of section overhead bytes and their values.

SDH/Sonet Path

- **SS bits**.
- **Path overhead bytes (POH)**.
- **Justification counters (JC)**.

DVB/ASI

These settings apply only for DVB/ASI type of interfaces:

- **Format**. Displays a message of the current video format.
- **Output mode**. Determines how MPEG TS packets are output.
    - **Burst**. All bytes of a TS packet are sent back-to-back in one burst. IDLE bytes are sent between bursts to adapt the bit stream.
    - **Spread**. Spread-byte mode. TS packet bytes are spread out as evenly as possible, with IDLE stuffing between TS packets as well as between bytes within a TS packet. This is the preferred mode.

© 2003-2010 Net Insight AB, All Rights Reserved

- **Auto**. Burst or Spread mode is automatically selected to be the same as detected on the corresponding ingress port.

The **Apply** button immediately applies the changes.

The **Ok** button immediately applies the changes and closes the window.

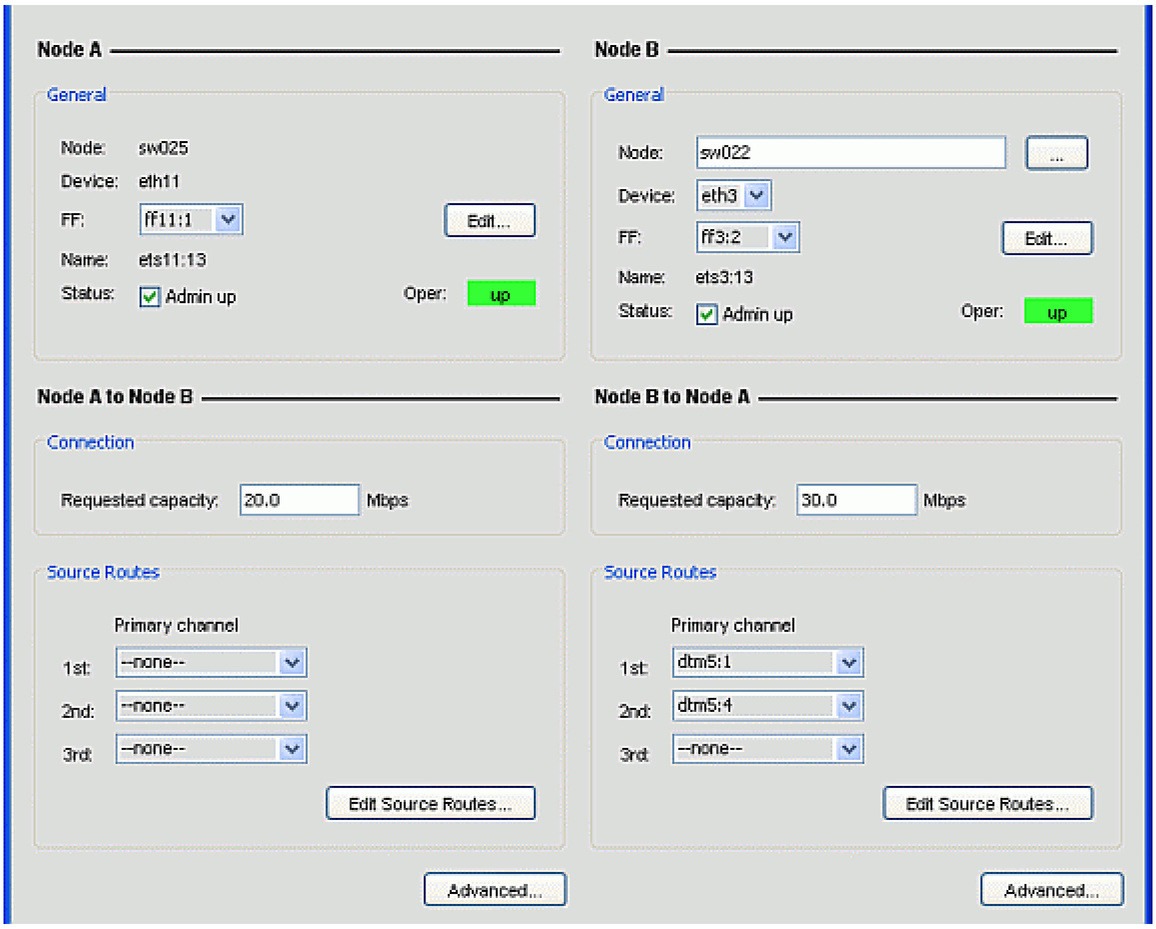
The **Cancel** button closes the windows without applying the changes.

Ethernet Transport Service, ETS

ETS Service Editor (unicast)

ETS unicast is a bi-directional service, i.e. a provisioned service has connections in both directions between the two peering nodes.

The top part of the Service Editor contains the **Administrative data** and the Global admin status control. This section is the same on all variants of the service editor. See section Editing a Service for the general description of this part.


Node A

General

Node: sw025

Device: eth11

FF: ff11:1

Name: ets11:13

Status: ✓ Admin up   Oper: up

Node B

General

Node: sw022

Device: eth3

FF: ff3:2

Name: ets3:13

Status: ✓ Admin up   Oper: up

Node A to Node B

Connection

Requested capacity: 20.0 Mbps

Source Routes

Primary channel

1st: --none--

2nd: --none--

3rd: --none--

Node B to Node A

Connection

Requested capacity: 30.0 Mbps

Source Routes

Primary channel

1st: dtm5:1

2nd: dtm5:4

3rd: --none--

The connection specific configuration of an ETS unicast service.

An ETS unicast service is represented by two services in Nimbra Vision, one for each peer. The left half of the Service Editor under the heading **Node A** represents the peer that was selected when the Service Editor was opened, or the source node when the service was created. The right half of the Service Editor

© 2003-2010 Net Insight AB, All Rights Reserved

under the heading **Node B** represents the other peer. Editing a unicast ETS service is therefore updating both services.

To complete the provisioning of an ETS service, the service must be associated with a physical Ethernet interface (ETSv1) or a forwarding function (ETSv2). For ETSv1, this association is done using VLANs. Use the Ethernet Configuration Editor to add and associate VLANs with the service and the interface. For ETSv2, the associated forwarding function forwards frames to other interfaces.

Node A

The **General** section contents depends on whether the node supports ETS version 1 or version 2.

- **Node** is the name of the node where the service originates. This is defined when the service is created, and cannot be changed.
- **Device** (ETSv2 only) is the name of the device on which the service is located. A service cannot be moved between different devices.
- **FF** (ETSv2 only) is the forwarding function to where the service is associated. To edit the selected forwarding function, click the **Edit…** button. This opens the Forwarding Function Editor. To create a new forwarding function, select `--new--` in the combo box. The **Edit…** button then becomes a **Create** button. Click to create.
- **Name** is the name of the TTP in the originating node. The name is automatically generated.
- **Admin up** controls the administrative state of the originating TTP. The normal way to control the administrative state of the service is to control all TTPs in a single operation using the **Set admin status to control** at the general part of the page, see chapter Editing a Service.
- **Oper** displays the operational state of the TTP.

Node A to Node B

The **Connection** section:

- **Requested capacity**. Specify the capacity in Mbps for the connection originating in this node.

The **Source Routes** section contains a list of source routes. A source route is a definition of a path through the network.

- **1st/2nd/3rd**: The rows represent what source route to use as first, second or third choice. If a source route is not specified, it is presented as `--none--`. The name of the currently used source route is marked with a tick. See chapter Source Routes for details.
- **Edit Source Routes…**. This button accesses dialogs for editing and displaying source routes on the node where the service originates. See chapter List of Source Routes.

Node B

The General section contains settings similar to the settings under heading Node A, but for the reverse direction. It is possible to change the node etc. here. Changing the node or device will deleted the returning service and create a new returning service.

The **Advanced…** button at the end of the Source part of the page opens a dialog for advanced settings, see chapter Advanced Settings for Source.

© 2003-2010 Net Insight AB, All Rights Reserved

The **Ethernet…** button (ETSv1 only) next to the **Advanced…** button opens dialogs for assigning of VLANs and Ethernet ports. See chapter Ethernet Configuration Editor.

See Also

Editing a Service for details common to all Service Editors.

Ethernet Configuration Editor for configuration of VLANs and Ethernet ports.

List of Source Routes for details on configuring and displaying source routes.

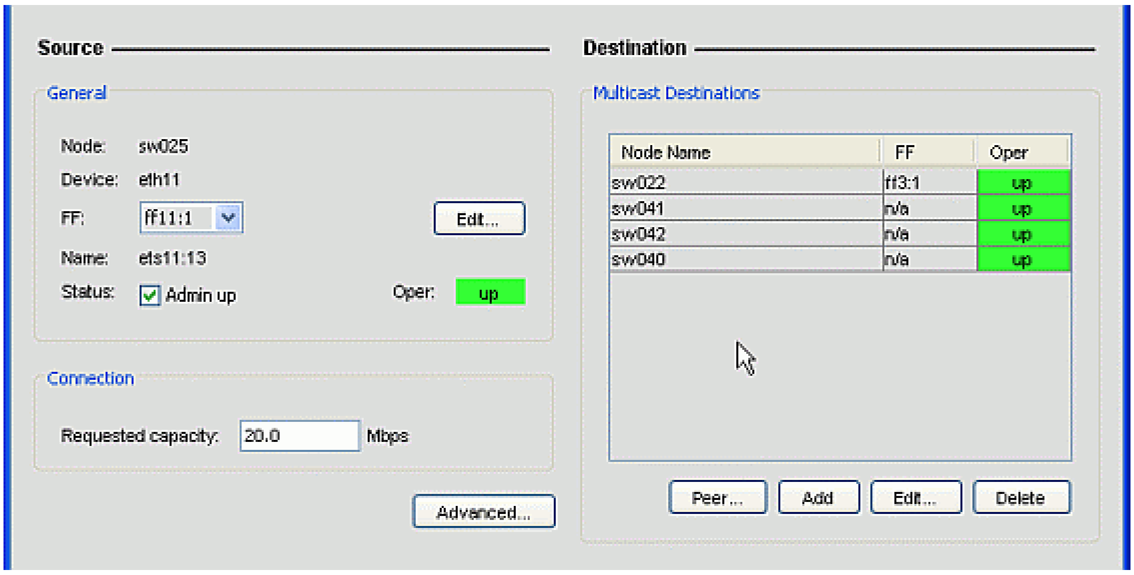
Source Routes for general description of source routes.

Advanced Settings for Source for details on how to configure advanced settings such as DSTI, precedence etc.

Advanced Settings for Destination for details on how to configure advances settings such as DSTI.

ETS Service Editor (multicast)

The top part of the Service Editor contains the **Administrative data** and the Global admin status control. This section is the same on all variants of the service editor. See section Editing a Service for the general description of this part.



Node:	sw025
Device:	eth11
FF:	ff11:1
Name:	ets11:13
Status:	✓ Admin up
Oper:	up
Requested capacity:	20.0 Mbps

Node Name	FF	Oper
sw022	ff3:1	up
sw041	n/a	up
sw042	n/a	up
sw040	n/a	up

The connection specific configuration of an ETS multicast service.

An ETS multicast service is represented by one service in each peer. ETS multicast can have connections originating and terminating in the same TTP, and it is possible to setup the service as a simple tree, as a full mesh, or as something in between. The multicast service is therefore always provisioned as a uni-directional broadcast tree starting at the **Source**, and with several **Destinations**. A meshed service must therefore be provisioned at separate services on all source nodes. The ETS Service Editor for multicast connections represents the multicast tree from the source node.

To complete the provisioning of an ETS service, the service must be associated with a physical Ethernet interface (ETSv1) or a forwarding function (ETSv2). For ETSv1, this association is done using VLANs. Use the Ethernet Configuration Editor to add and associate VLANs with the service and the interface. For ETSv2, the associated forwarding function forwards frames to other interfaces.

Source

The **General** section:

© 2003-2010 Net Insight AB, All Rights Reserved

- **Node** is the name of the node where the service originates. This is defined when the service is created, and cannot be changed.
- **Device** (ETSv2 only) is the name of the device on which the service is located. A service cannot be moved between different devices.
- **FF** (ETSv2 only) is the forwarding function to where the service is associated. To edit the selected forwarding function, click the **Edit…** button. This opens the Forwarding Function Editor. To create a new forwarding function, select `--new--` in the combo box. The **Edit…** button then becomes a **Create** button. Click to create.
- **Name** is the name of the TTP in the originating node. The name is automatically generated.
- **Admin up** controls the administrative state of the originating TTP. The normal way to control the administrative state of the service is to control all TTPs in a single operation using the **Set admin status to control** at the general part of the page, see chapter Editing a Service.
- **Oper** displays the operational state of the TTP.

The **Connection** section:

- **Requested capacity**. Specify the capacity in Mbps for the connection.

The **Source Routes** section contains a list of source routes. A source route is a definition of a path through the network.

- **1st/2nd/3rd**: The rows represent what source route to use as first, second or third choice. If establishment of a channel using the first source route fails, then the node will try the second, and so on.
- **Edit Source Routes…**. This button accesses dialogs for editing and displaying source routes on the node where the service originates. See chapter List of Source Routes.

The **Advanced…** button at the end of the Source part of the page opens a dialog for advanced settings, see chapter Advanced Settings for Source.

Destination

The **Multicast Destinations** section displays the multicast destinations. The section contains a table where each destination is represented by a row. The table displays the node and the operational state. Below the table are the following buttons:

- **Peer**. Opens a new Service Editor for the selected destination. You can use this button as a shortcut to quickly open a Service Editor for configuration of the service seen as originating on the peering node.
- **Add**. Opens a dialog for adding a destination. See chapter ETS Multicast Destination.
- **Edit**. Opens a dialog for editing the destination selected in the table. See chapter ETS Multicast Destination.
- **Delete**. Deletes the destination selected in the table.

Configuration of Bidirectional ETS Multicast

ETS multicast can be used as a unidirectional or bidirectional service. ETS multicast connections are always configured as unidirectional connections, with a source and several destinations. To use the service as bidirectional, you must configure a return connection from the destination.

The service makes no restrictions on how connections are configured. A multicasts TTP can act as a source TTP having one multicast connection to

© 2003-2010 Net Insight AB, All Rights Reserved

several nodes, and it can act as a destination TTP terminate many multicast connections. It is recommended to keep the relationships as simple as possible.

When you configure bidirectional multicast connections, you will configure the return connection to an already existing destination. You should not create new destinations; instead use the already existing source as destination. This is because the destination is actually also the source as seen from the other direction.

To setup the return from B to A, do:

1. Open the Service Editor for the service originating in node A.
2. In that Service Editor, open the **Advanced…** dialog and make a note of the DSTI assigned to the TTP. Close the dialog.
3. Open the Service Editor for the service originating in node B. A simple way to do this is to select the destination B in the Destinations table and click the button **Peer…**.
4. In that Service Editor, add a new destination to A. In the Multicast Destination dialog, open the **Advanced…** dialog. Select the **Use another service's destination with DSTI**, and specify the previously noted DSTI. Click **OK**, this closes the dialog.
5. Click **OK** in the Multicast Destination dialog. This closes the dialog.
6. Click **OK/Apply** in the Service Editor. This provisions a connection to the destination.

See Also

Editing a Service for details common to all Service Editors.

Ethernet Configuration Editor for configuration of VLANs and Ethernet ports.

List of Source Routes for details on configuring and displaying source routes.

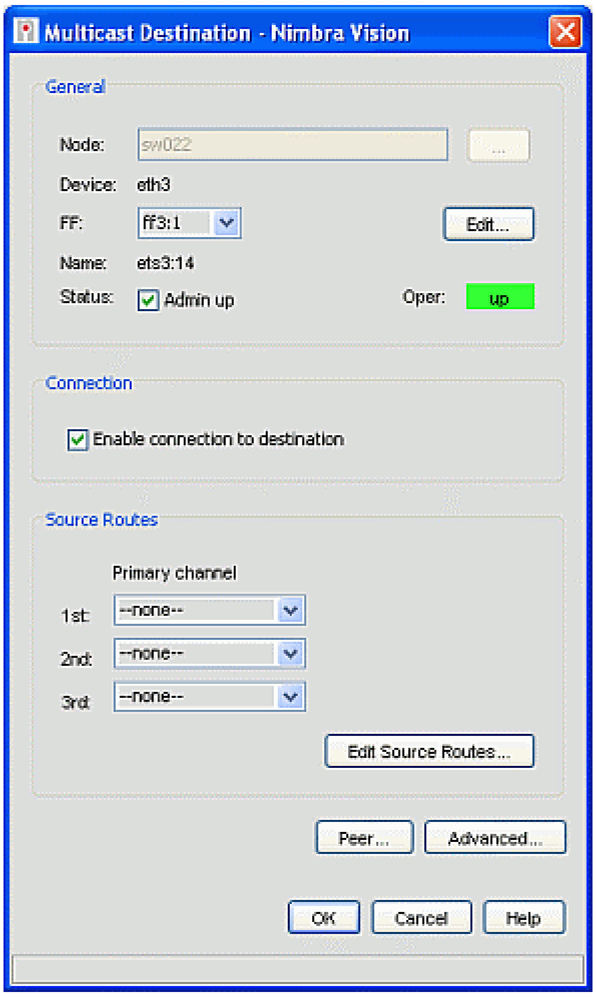
Advanced Settings for Source for details on how to configure advanced settings such as DSTI, precedence etc.

ETS Multicast Destination.

ETS Multicast Destination

This dialog describes a single destination for an ETS multicast service. It is similar to the **Node B** part in the Service Editor for a unicast service. The dialog is opened from the Service Editor.

© 2003-2010 Net Insight AB, All Rights Reserved


Multicast Destination - Nimbra Vision

General

Node: sw022

Device: eth3

FF: ff3:1

Name: ets3:14

Status: ✓ Admin up

Oper: up

Connection

✓ Enable connection to destination

Source Routes

Primary channel

1st:	--none--
2nd:	--none--
3rd:	--none--

ETS multicast destination dialog.

The **General** section:

- **Node** is the name of the node where the service terminates. When applied, changing of the node will delete the previous TTP and create a new TTP on the new node.
- **Device** (ETSv2 only) is the name of the device on which the service is located. When applied, changing the device will delete the previous TTP and create a new TTP on the new device.
- **FF** (ETSv2 only) is the forwarding function to where the service is associated. To edit the selected forwarding function, click the **Edit…** button. This opens the Forwarding Function Editor. To create a new forwarding function, select `--new--` in the combo box. The **Edit…** button then becomes a **Create** button. Click to create.
- **Name** is the name of the TTP in the originating node. The name is automatically generated.
- **Admin up** controls the administrative state of the originating TTP. The normal way to control the administrative state of the service is to control all TTPs in a single operation using the **Set admin status to control** at the general part of the page, see chapter Editing a Service.
- **Oper** displays the operational state of the TTP.

The **Connection** section:

- **Enable connection to destination**. The checkbox controls whether the connection shall be established to the destination or not. Using this, you

© 2003-2010 Net Insight AB, All Rights Reserved

can disable the connection to individual multicast destinations, but still keep the configuration to the destination for future use.

The **Source Routes** allows for assigning source routes to the connection. A source route is a definition of a path through the network:

- **1st/2nd/3rd**: The rows represent what source route to use as first, second or third choice. If a source route is not specified, it is presented as `--none--`. The name of the currently used source route is marked with a tick. See chapter Source Routes for details.
- **Edit Source Routes…**. This button accesses dialogs for editing and displaying source routes on the node where the service originates. See chapter List of Source Routes.

The **Advanced…** button opens a dialog for advanced settings, e.g. setting of DSTI etc., see chapter Advanced Settings for Destination.

The **Ethernet…** button (ETSv1 only) next to the **Advanced…** button opens dialogs for assigning of VLANs and Ethernet ports. See chapter Ethernet Configuration Editor.

See Also

ETS Service Editor (multicast)

Advanced Settings for Destination for details on how to configure advances settings such as DSTI.

List of Source Routes for details on configuring and displaying source routes.

Source Routes for general description of source routes.

Forwarding Function (ETSv2)

The Forwarding Function Editor allows configuration of the forwarding function. The editor has the following tabs:

- General
- Interfaces
- DiffServ
- VLAN
- RSTP

The buttons at the bottom of the editor:

Click button **Refresh** to read all the current settings of the forwarding functioin from the node and updated the editor with the data.

Click button **OK** to save the settings to the node, and close the window.

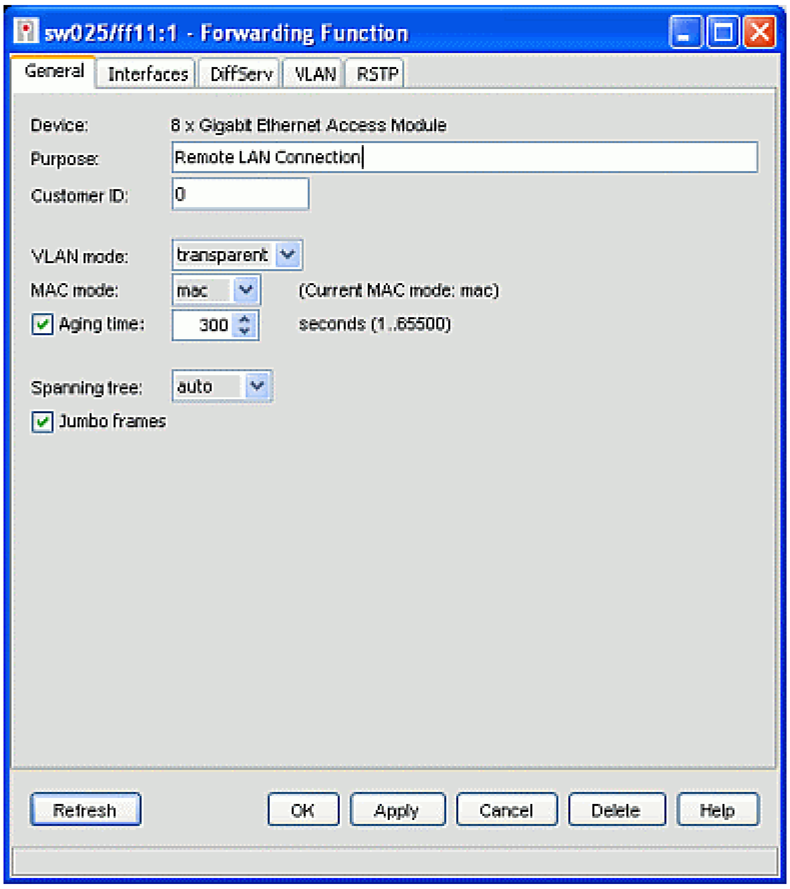
Click button **Apply** to save the setting to the node, and keep the window open.

Click button **Cancel** to discard any unsaved settings and close the window.

Click button **Help** to display help for the displayed tab.

© 2003-2010 Net Insight AB, All Rights Reserved

General


sw025/ff11:1 - Forwarding Function

Device: 8 x Gigabit Ethernet Access Module
Purpose: Remote LAN Connection
Customer ID: 0
VLAN mode: transparent
MAC mode: mac (Current MAC mode: mac)
✓ Aging time: 300 seconds (1..65500)
Spanning tree: auto
✓ Jumbo frames

Forwarding function general tab.

- **Device**. Displays the type of device that the forwarding function is a part of.
- **Purpose**. A free text string that can be used for describing the purpose or usage of the forwarding function.
- **Customer ID**. A numerical value representing a customer.
- **VLAN Mode**.
    - **transparent**. Ignore VLAN tags on received frames. VLAN tags are not added nor removed from the frames. If a frame has a VLAN tag, then it is retained. If a frame does not have a VLAN tag, then no tag is added. If incoming frames are tagged, then **Jumbo frames** may have to be enabled to prevent too large frames to be discarded.
    - **customer**. VLAN-handling in accordance with IEEE 802.1Q. Frames are forwarded and filtered according to customer VLAN tag. Frames that are not tagged are tagged with default VLAN tag as configured on the interface.
    - **provider**.
- **MAC mode**.
    - **auto**. Automatically select a good setting: equivalent to *mac* if when the forwarding function is attached to at least three interfaces, otherwise equivalent to *nomac*.
    - **mac**. The forwarding function will learn what interfaces to use for a destination. This is done by looking at the source MAC address on received frames. When the forwarding

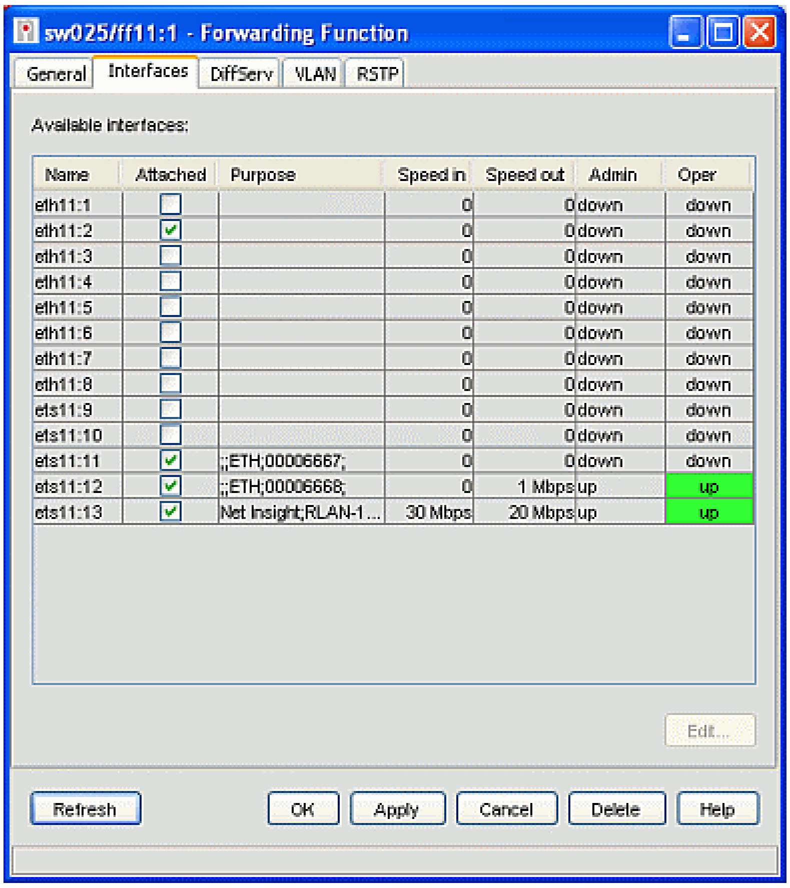
© 2003-2010 Net Insight AB, All Rights Reserved

function has learned what interface to use, it will forward the frame on that interface only. Learning must also be enabled on the interface.

 - ○ **nomac**. The forwarding function will forward the frames on all interfaces, except on the interface where the frame arrived.

- **Aging time**. Timeout for aging out learned forwarding information when **MAC mode** is set to *mac*.
- **Spanning tree**.
  - ○ **auto**. Automatically select a good setting: equivalent to *forward* when exactly two interfaces are attached to the forwarding function, otherwise equivalent to *process*.
  - ○ **forward**. Receive spanning tree messages are broadcasted to all interface except the interface where the message was received. No further processing is done.
  - ○ **drop**. Received spanning tree messages are discarded. No further processing is done.
  - ○ **process**. Received spanning tree messages are processed according the spanning tree protocol and configuration.
- **Jumbo frames**. Enable or disable forwarding of Jumbo frames.

Interfaces



Available interfaces:

Name	Attached	Purpose	Speed in	Speed out	Admin	Oper
eth11:1	☐		0	0	down	down
eth11:2	✓		0	0	down	down
eth11:3	☐		0	0	down	down
eth11:4	☐		0	0	down	down
eth11:5	☐		0	0	down	down
eth11:6	☐		0	0	down	down
eth11:7	☐		0	0	down	down
eth11:8	☐		0	0	down	down
ets11:9	☐		0	0	down	down
ets11:10	☐		0	0	down	down
ets11:11	✓	;;ETH;00006667;	0	0	down	down
ets11:12	✓	;;ETH;00006668;	0	1 Mbps	up	up
ets11:13	✓	Net Insight;RLAN-1 ...	30 Mbps	20 Mbps	up	up

Forwarding function interfaces tab.

The tab displays all interfaces that are attached to the forwarding function, or that can be attached to the forwarding function. Interfaces that are attached to other forwarding functions are not displayed in this table. The table contains

© 2003-2010 Net Insight AB, All Rights Reserved

both physical interfaces and TTPs. The forwarding function forwards frames between the attached interfaces.

To open an Ethernet interface editor for an interface, select the interface and click the button **Edit…**.

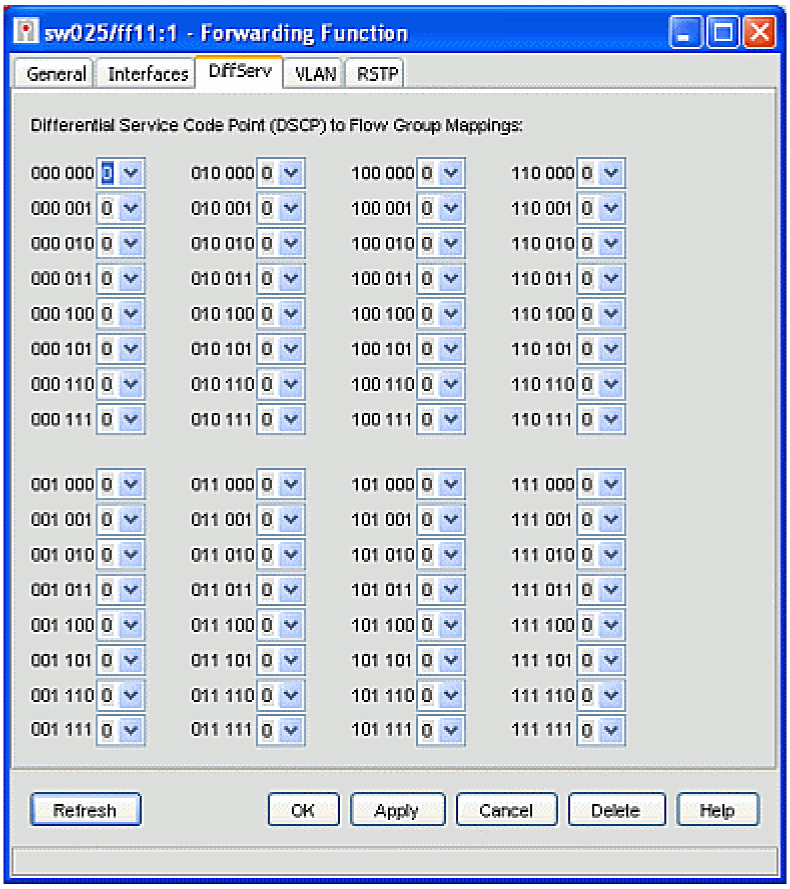
The table contains the following data:

- **Name**. The name of the interface.
- **Attached**. The checkbox is checked for interfaces that are attached to the forwarding function.
- **Purpose**. The purpose text string.
- **Speed in**. The input speed of the interface.
- **Speed out**. The output speed of the interface.
- **Admin**. The administrative state of the interface.
- **Oper**. The operational state of the interface.

See Also

Ethernet Interface (ETSv2)

DiffServ



Differential Service Code Point (DSCP) to Flow Group Mappings:

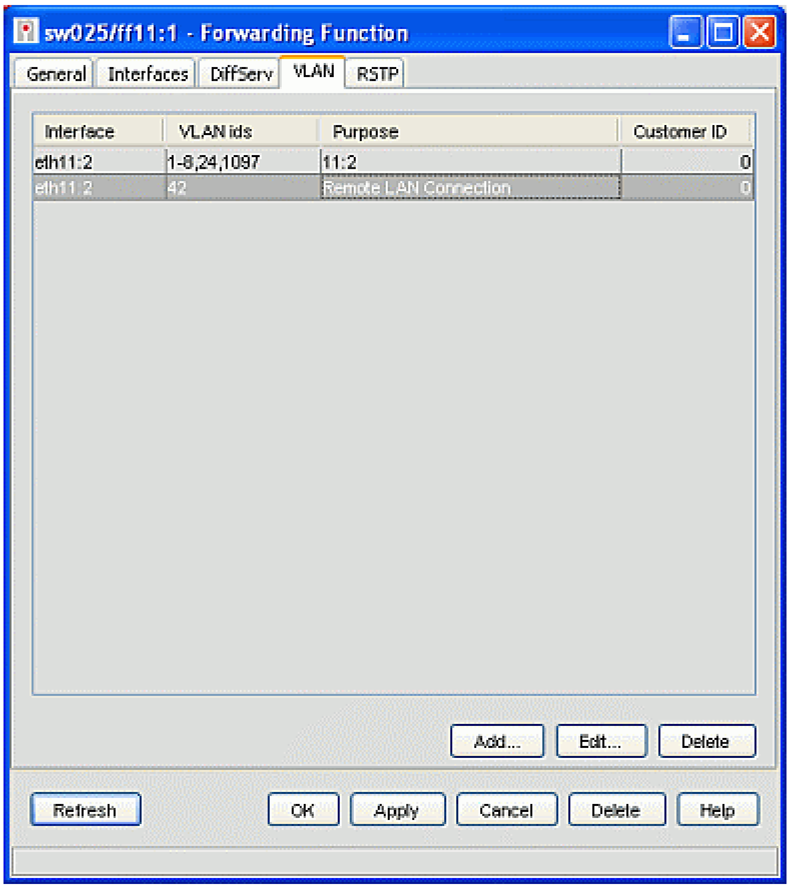
000 000	0	010 000	0	100 000	0	110 000	0
000 001	0	010 001	0	100 001	0	110 001	0
000 010	0	010 010	0	100 010	0	110 010	0
000 011	0	010 011	0	100 011	0	110 011	0
000 100	0	010 100	0	100 100	0	110 100	0
000 101	0	010 101	0	100 101	0	110 101	0
000 110	0	010 110	0	100 110	0	110 110	0
000 111	0	010 111	0	100 111	0	110 111	0
001 000	0	011 000	0	101 000	0	111 000	0
001 001	0	011 001	0	101 001	0	111 001	0
001 010	0	011 010	0	101 010	0	111 010	0
001 011	0	011 011	0	101 011	0	111 011	0
001 100	0	011 100	0	101 100	0	111 100	0
001 101	0	011 101	0	101 101	0	111 101	0
001 110	0	011 110	0	101 110	0	111 110	0
001 111	0	011 111	0	101 111	0	111 111	0

Forwarding Function Differential Services tab.

The tab defined the mapping between the Differential Services Code Points (DSCP) and the Flow Groups. The page displays a matrix with all 64 DSCP's. For each DSCP, a combo-box allows for mapping to a Flow Group.

© 2003-2010 Net Insight AB, All Rights Reserved

VLAN



Interface	VLAN ids	Purpose	Customer ID
eth11:2	1-8,24,1097	11:2	0
eth11:2	42	Remote LAN Connection	0

Forwarding Function VLAN tab.

The tab displays all VLAN sets on interfaces attached to this forwarding function.

A VLAN set is a list of VLAN ids.

The following buttons modifies the table:

Use button **Add...** to open the VLAN editor and adding a new VLAN set to an attached interface.

Use button **Edit...** to edit a selected VLAN set.

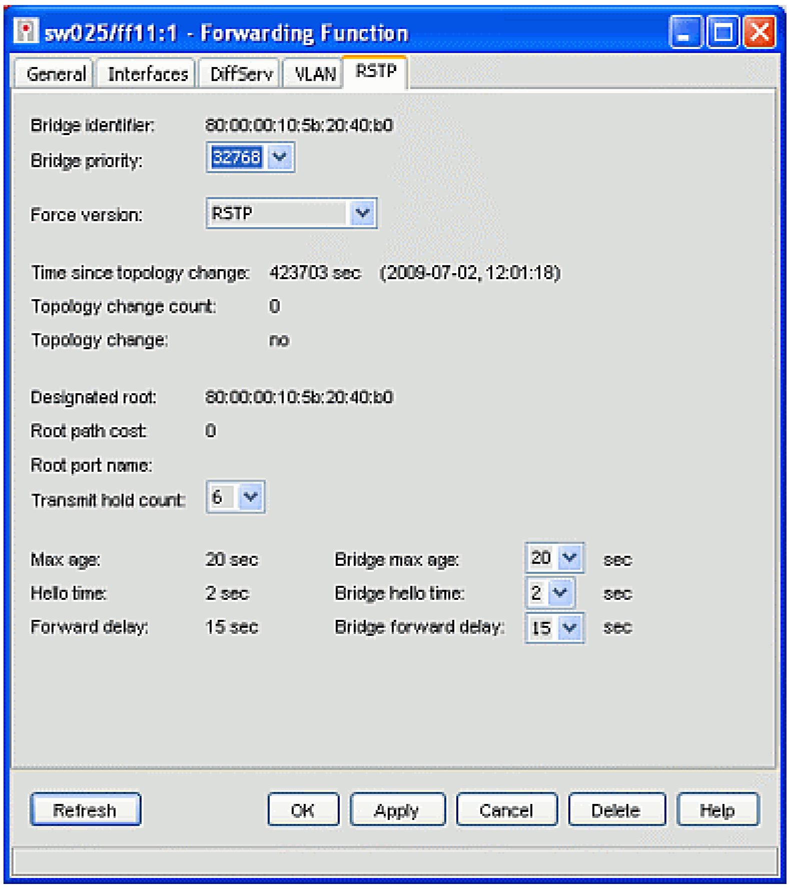
Use button **Delete** to delete a selected VLAN set.

See Also

Edit VLANs (ETSv2)

© 2003-2010 Net Insight AB, All Rights Reserved

RSTP



sw025/ff11:1 - Forwarding Function

Bridge identifier: 80:00:00:10:5b:20:40:b0
Bridge priority: 32768
Force version: RSTP
Time since topology change: 423703 sec (2009-07-02, 12:01:18)
Topology change count: 0
Topology change: no
Designated root: 80:00:00:10:5b:20:40:b0
Root path cost: 0
Root port name:
Transmit hold count: 6

Max age:	20 sec	Bridge max age:	20	sec
Hello time:	2 sec	Bridge hello time:	2	sec
Forward delay:	15 sec	Bridge forward delay:	15	sec

Forwarding function RSTP tab.

The tab contains setting for the Spanning Tree protocol. For details on the spanning tree protocol, see IEEE 802.1D-2004.

Ethernet Interface (ETSv2)

The Ethernet Interface Editor allows configuration of the interface. The same editor is used for physical interfaces and virtual interfaces. A virtual interface is the TTP where a service connection originates or terminates.

The Ethernet Interface Editor contains the following tabs:

- Basic
- Advanced
- VLAN
- RSTP
- AutoNeg

The buttons at the bottom of the editor:

Click button **Refresh** to read all the current settings of the interface from the node and updated the editor with the data.

Click button **OK** to save the settings to the node, and close the window.

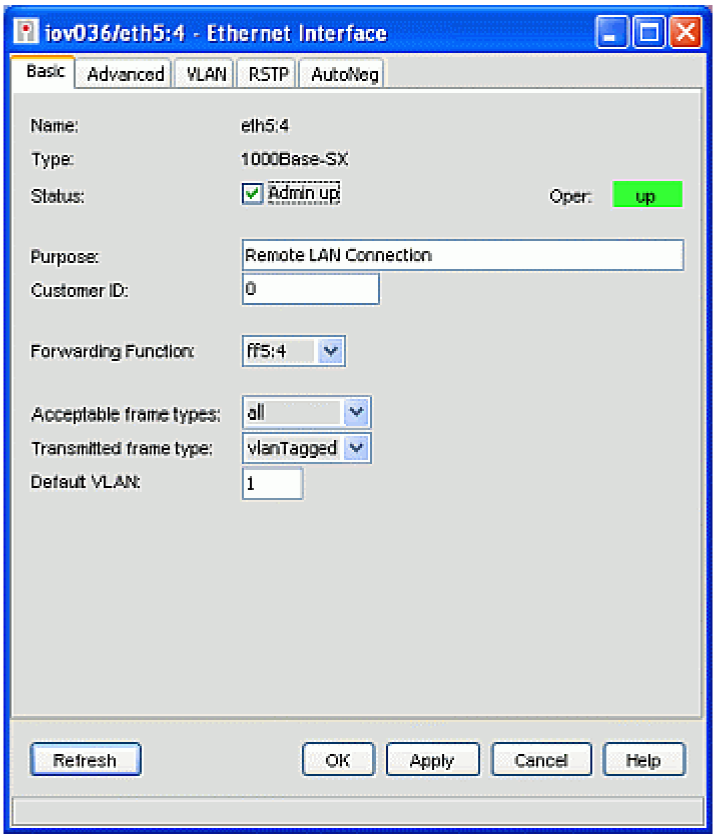
Click button **Apply** to save the setting to the node, and keep the window open.

Click button **Cancel** to discard any unsaved settings and close the window.

Click button **Help** to display help for the displayed tab.

© 2003-2010 Net Insight AB, All Rights Reserved

Basic



Name:	eth5:4
Type:	1000Base-SX
Status:	✓ Admin up — Oper: up
Purpose:	Remote LAN Connection
Customer ID:	0
Forwarding Function:	ff5:4
Acceptable frame types:	all
Transmitted frame type:	vlanTagged
Default VLAN:	1

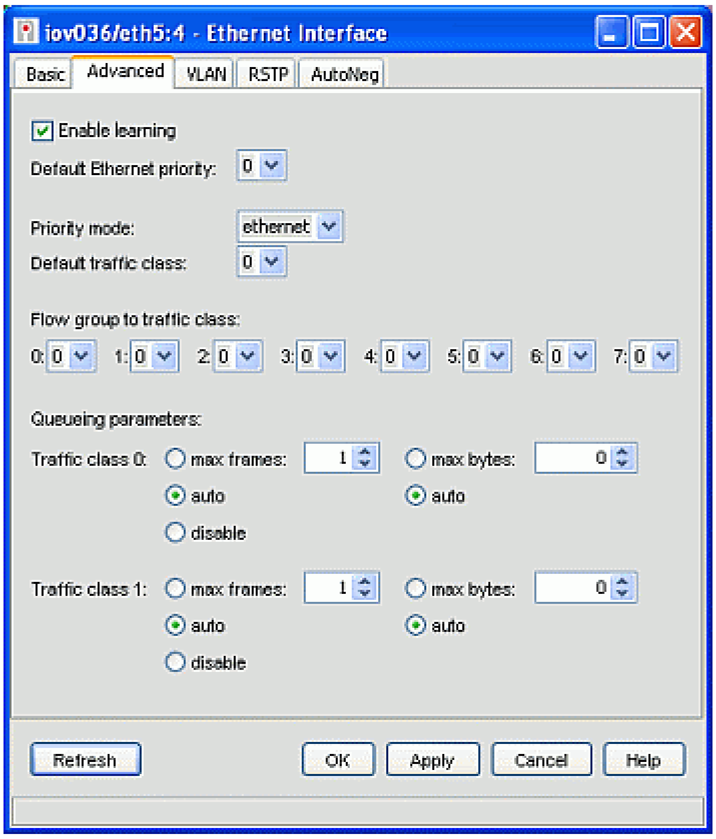
Ethernet interface basic tab.

- **Name**. The interface name within the node. The name of physical interfaces starts with the `eth`, while virtual interfaces starts with `ets`. The name then contains two number separated by a colon. The first number is the board number, the second number is the port number within the board.
- **Type**. The type of interface. Virtual interfaces have the type ETS.
- **Status**. The operational and administrative state of the interface.
- **Purpose**. A text field.
- **Customer ID**. A number representing a customer.
- **Forwarding Function**. The forwarding function to where this interface is attached.
- **Acceptable frame types**. Specifies how to accept frames. Note that provider or customer tags are examined, or are all ignored, depending on the attached forwarding functions VLAN mode setting.
  - **all**. All frames received on the interface are accepted.
  - **vlanTagged**. Only frames containing a non-zero VLAN id are accepted.
  - **untagged**. Only frames without a VLAN tag, or with VLAN id of zero are accepted.
- **Transmitted frame types**. Specifies the general setting on how forwarded (transmitted) frames are being tagged. Note that provider or customer tags are examined, or are all ignored, depending on the attached forwarding functions VLAN mode setting.

© 2003-2010 Net Insight AB, All Rights Reserved

  - **vlanTagged**. The frame is tagged. You can force frames to be untagged for specified VLAN ids in tab VLAN.
  - **untagged**. Any tag is removed from the frame. You force frames to be tagged for specified VLAN ids in tab VLAN.
  - **legacy**. The VLAN id is transported with the frame, but the frame itself is not tagged. This is applicable only for virtual interfaces (ETS), and its use together with ETSv1.
- **Default VLAN**. If an untagged frame is received on the interface, then it will be tagged with the specified VLAN id. The VLAN id is a customer or provided tag, or ignored, depending on the attached forwarding functions VLAN mode setting.

Advanced


iov036/eth5:4 - Ethernet Interface

Basic | Advanced | VLAN | RSTP | AutoNeg

☑ Enable learning

Default Ethernet priority: 0

Priority mode: ethernet

Default traffic class: 0

Flow group to traffic class:

0: 0 1: 0 2: 0 3: 0 4: 0 5: 0 6: 0 7: 0

Queueing parameters:

Traffic class 0: ○ max frames: 1 ○ max bytes: 0
● auto ● auto
○ disable

Traffic class 1: ○ max frames: 1 ○ max bytes: 0
● auto ● auto
○ disable

Ethernet interface advanced tab.

- **Enable learning**. Enable of disable MAC learning. When MAC learning is enabled, then the attached forwarding function can learn what MAC addresses to use for a destination by looking at the source MAC address in received frames. The value is only used when **MAC mode** is set to *mac* on the attached forwarding function.
- **Default Ethernet priority**. The Ethernet priority (between 0 and 7) being assigned to an arriving frame that does not already have this information in its header (this is in the VLAN tag). The priority is only assigned to the frame if it should be sent out tagged. The default Ethernet priority does not affect the traffic class being assigned to the frame; the traffic class is assigned according to **Priority mode** setting.
- **Priority mode**. The method used when assigning a flow group to a frame.

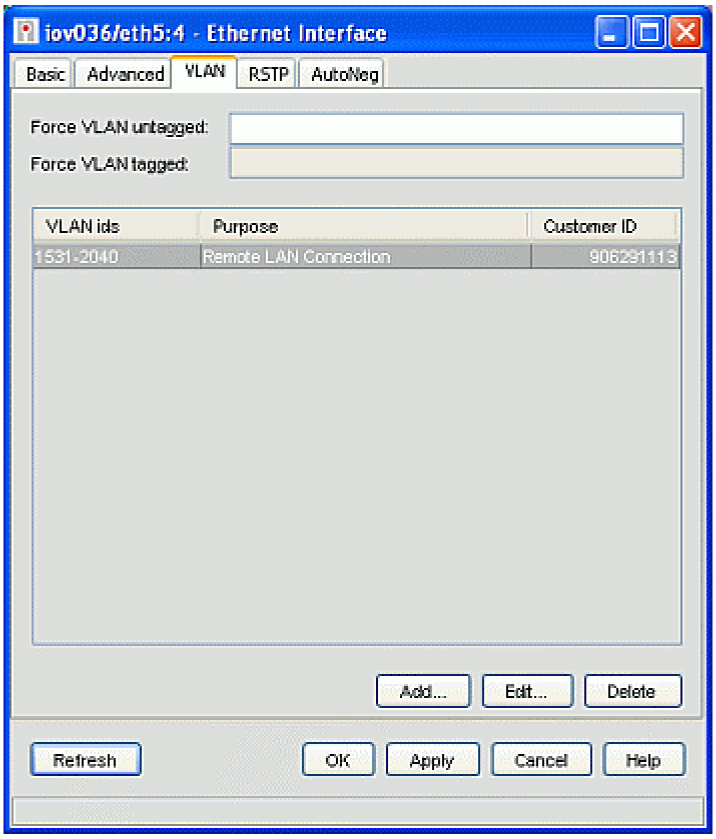
© 2003-2010 Net Insight AB, All Rights Reserved

    - **ethernet**. The Ethernet priority user bits are used. If the received frame does not have any priority information, then the **default traffic class** is assigned.
    - **diffserv**. The differential service code point (DSCP) field of the IP header is translated to a flow group using the Differential Service Code Point to Flow Group mapping configured on the forwarding function. (see DiffServ tab on Forwarding Function).

- **Default traffic class**. The traffic class to assign frames that do not have any priority information. Traffic class 0 has the lowest priority. Typically, there are only the two traffic classes 0 (low priority) and 1 (high priority).
- **Flow group to traffic class**. Mapping of the eight flow groups into the (two) traffic classes.
- **Queuing parameters**. Configuration of the queues for the traffic classes. Queuing parameters are configured per traffic class:
    - **max frames**. The maximum number of frames that can be queued for input on the traffic class. If the queue is full when a frame arrives, then it is dropped. If the setting is **auto**, then the queue length is automatically assigned based on the number of available buffers. If the setting is **disable**, then queuing is disabled and all frames are dropped.
    - **max bytes**. The maximum number of bytes that can be queued for input on the traffic class. If an arriving frame is too large to fit into the queue, then it is dropped. If the setting is **auto**, then the queue setting is automatically set to ensure that the queuing delay is less than one second.

© 2003-2010 Net Insight AB, All Rights Reserved

VLAN



Force VLAN untagged:

Force VLAN tagged:

VLAN ids	Purpose	Customer ID
1531-2040	Remote LAN Connection	906291113

Ethernet interface VLAN tab.

The tab contains a table with all VLAN sets on this interface.

- **Force VLAN untagged**. Specifies a set of VLAN ids for which frames shall be forwarded without a tag although **Transmitted frame type** in tab Basic is set to *vlanTagged*.
- **Force VLAN tagged**. Specifies a set of VLAN ids for which frames shall be forwarded with a tag although **Transmitted frame type** in tab **Basic** is set to *untagged*.

The following buttons modifies the table:

Click button **Add…** to open the VLAN editor and adding a new VLAN set to the interface.

Click button **Edit…** to edit a selected VLAN set.

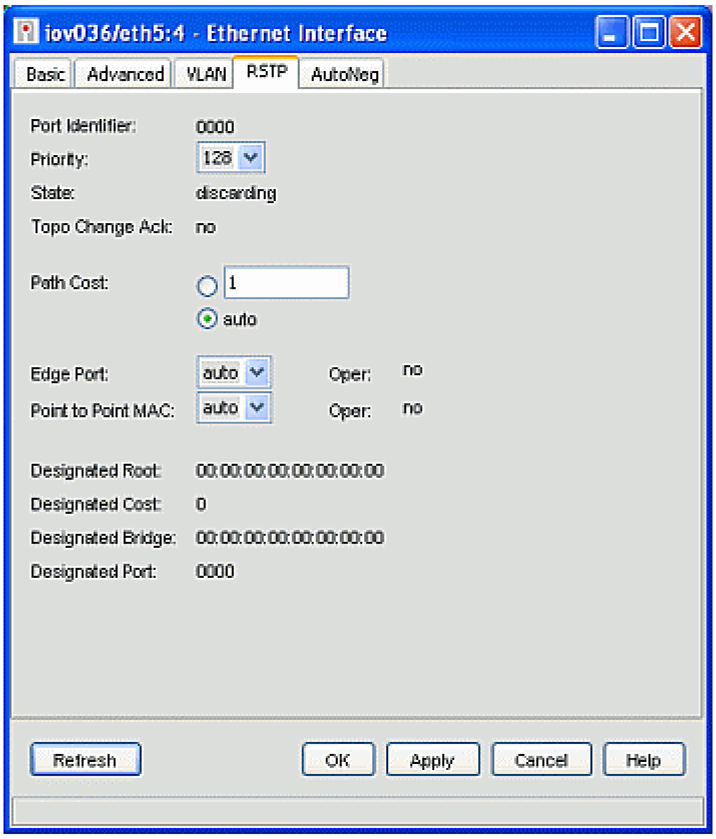
Click button **Delete** to delete a selected VLAN set.

See Also

Edit VLANs (ETSv2)

© 2003-2010 Net Insight AB, All Rights Reserved

RSTP



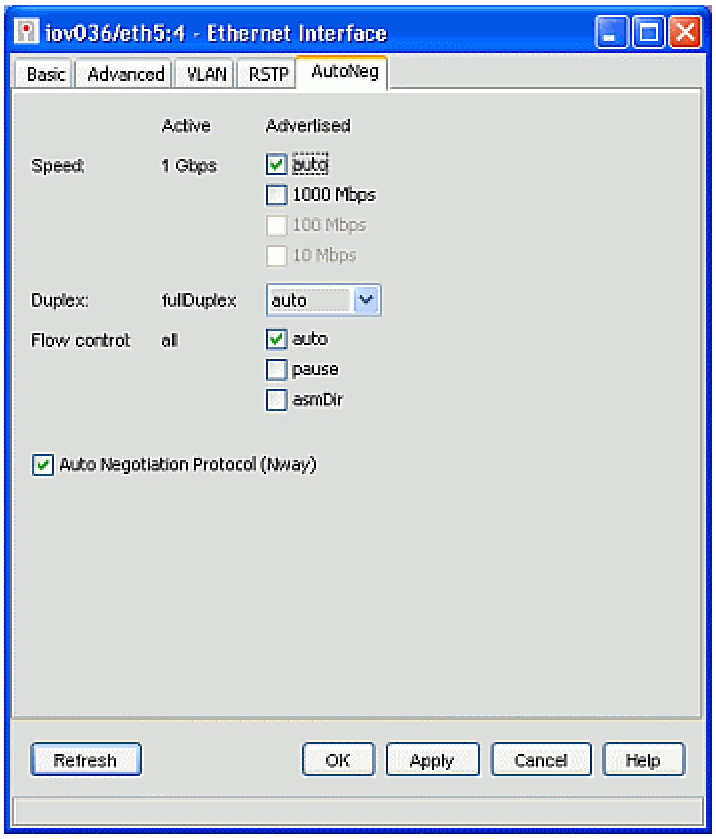
Port Identifier:	0000		
Priority:	128		
State:	discarding		
Topo Change Ack:	no		
Path Cost:	○ 1 / [✓ auto]		
Edge Port:	auto	Oper:	no
Point to Point MAC:	auto	Oper:	no
Designated Root:	00:00:00:00:00:00:00:00		
Designated Cost:	0		
Designated Bridge:	00:00:00:00:00:00:00:00		
Designated Port:	0000		

Ethernet interface RSTP tab.

The tab contains setting for the Spanning Tree protocol. For details on the spanning tree protocol, see IEEE 802.1D-2004.

© 2003-2010 Net Insight AB, All Rights Reserved

AutoNeg



	Active	Advertised
Speed:	1 Gbps	[✓] auto [ ] 1000 Mbps [ ] 100 Mbps [ ] 10 Mbps
Duplex:	fullDuplex	auto
Flow control:	all	[✓] auto [ ] pause [ ] asmDir

[✓] Auto Negotiation Protocol (Nway)

Ethernet interface auto-negotiation tab

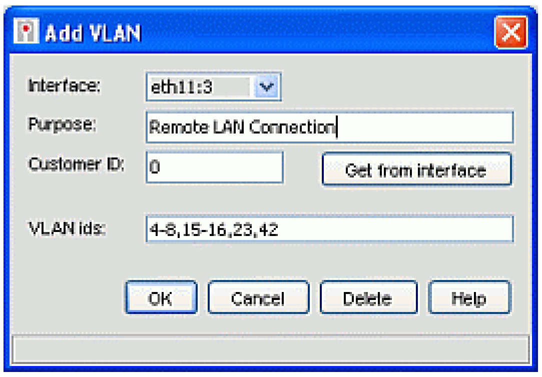
The tab contains settings for auto-negotiation. This tab is only available for physical interfaces.

- **Speed**. The active and advertised link speed. Advertised speed of **auto** advertises all speeds that the interface supports. If auto negotiation is disabled, then **auto** means that the highest supported speed is used.
- **Duplex**. The active and advertised duplex. Advertised duplex of **auto** advertises all duplex settings that the interface supports. If auto negotiation is disabled, the auto means that the maximum supported duplex is used.
- **Flow control**. The active and advertised flow control. Advertised flow control of **auto** advertises all setting supported by the interface. If auto negotiation is disabled, then **auto** means that flow control is enabled in both directions.
- **Auto Negotiation Protocol (Nway)**. Enable or disabled the auto-negotiation.

Edit VLANs (ETSv2)

Add or edit a VLAN set. A VLAN set contains a set of VLAN ids. The VLAN set makes it possible to specify administrative data together with the specified VLAN ids.

© 2003-2010 Net Insight AB, All Rights Reserved



Interface: eth11:3
Purpose: Remote LAN Connection
Customer ID: 0
VLAN ids: 4-8,15-16,23,42

Add/Edit VLAN set.

- **Interface**. Select the interface to which the VLAN set is to be used.
- **Purpose**. A text field.
- **Customer ID**. A number representing a customer
- **VLAN ids**. A comma (",") separated list of VLAN ids or VLAN id ranges. A range is specified as the first and last VLAN id in the range separated with a dash ("-"). You can enter VLAN ids or ranges in any order, even overlapping or repeated VLAN ids. Whenever the focus exits the text box, an optimized and sorted representation of the VLAN ids is calculated.

Click button **Get from interface** to copy the purpose and customer ID from the specified interface.

Click button **OK** to apply the changes and close the window.

Click button **Cancel** to discard the changes and close the dialog.

Click button **Delete** to delete the VLAN set and close the dialog.

VLAN Associations (ETSv1)

This dialog is only applicable for nodes supporting ETSv1.

This dialog can be opened either from an ETS service, or from the VLAN Editor. Use this dialog to associate a physical Ethernet port with a Service. The association is done with VLANs.

- **Purpose** This is administrative data similar to the same in the Service Editor.
- **Customer** This is administrative data similar to the same in the Service Editor.
- **Set Purpose and Customer to same as associated service**. If you tick this checkbox, then the Purpose and Customer will be updated to the same data as configured in the Service Editor for the service.
- **VLAN id**. This is the actual VLAN id.

The table contains all physical interfaces on the board. The table has the following columns:

- **Interface**, this is the name of the physical Ethernet interface.
- **Use interface**, tick the checkbox to associate the VLAN with this physical interface.
- **Tag packets leaving the interface**; tick this checkbox if all Ethernet packets leaving this interface shall keep the VLAN id. If the checkbox is un-ticked, then the VLAN id is removed from the packets. All Ethernet packets transmitted in ETS have a VLAN id. Packets received

© 2003-2010 Net Insight AB, All Rights Reserved

that do not have a VLAN id are assigned a default VLAN id. Configure the default VLAN id from the **Ethernet Interface** Editor.

To configure the physical Ethernet interface, click the button **Interface…**. It will open the Ethernet Interface Editor.

You can associate the VLAN with another service. To do this, select an interface in the table and click the **Advanced…** button. From here you can associate the VLAN with another service by selecting the service, or by its DSTI.

See Also

Ethernet Interfaces

Ethernet Configuration Editor (ETSv1)

This dialog is only applicable for nodes supporting ETSv1.

This dialog shows VLANs, and how they are associated with a physical interface.

If the dialog is opened form the Service Editor, then it will list all VLANs associated with the service from where it is opened.

If the dialog is opened from a selected node in the map or network database, then all VLANs are listed. In this case, you can filter to list only VLANs that exist on a specific board using the **View filter**.

The list is a table summarizing the association for each VLAN.

- **VLAN id**
- **Interface**, then name of the physical Ethernet interface.
- **Tagged**, ticked if packets leaving the interface keep the VLAN tags.
- **Purpose**, administrative data aiding in identifying the VLAN.

Buttons:

- **Add…** adds a new VLAN to a board, see below.
- **Edit…** opens a dialog to edit the VLAN associations for the selected VLAN.
- **Delete** deletes the selected VLAN.

Adding a VLAN

Click button **Add…** to add a new VLAN. The VLAN must be associated with a specific board. The button opens a wizard where you must select the board. In this dialog, select **Next>>** to open the VLAN Associations with new VLAN.

See Also

VLAN Associations

Ethernet Interface (ETSv1)

This dialog is only applicable for nodes supporting ETSv1.

This dialog allows for configuration of an Ethernet Interface. It is opened from the VLAN Associations dialog.

All Ethernet packets transmitted using ETS must be tagged with a VLAN id. If a packet received on the Ethernet interface is not tagged with a VLAN id, then the packet must be dropped, or the packet is tagged with a default VLAN id. This is configured with the radio buttons **For packets without VLAN id**:

- **Drop packet**. The packet is dropped.

© 2003-2010 Net Insight AB, All Rights Reserved

- **Tag with VLAN id:** The packet is tagged with the selected VLAN id. You can only select an already existing VLAN.

See Also

VLAN Associations

Network Functions

Channel Persistence

Given a large network, it is inevitable that failures occur from time to time. These failures can affect transmission links as well as switching nodes. When an error occurs somewhere in a DTM network, the default action is to tear down all channels that are affected by the error. The reason for this is that the network can hopefully find another path through the network for the channel and restore the service quickly.

Tearing down the channels is however not always the best course of action. In some situations, channels can continue to forward data even though a node controller has stopped working or a link has become uni-directional. In other situations, it might be appropriate to let a channel remain established even though a link that it is running over has failed. This will lead to faster recovery when the link is repaired. The Channel Persistence functionality allows the operator to configure the behavior of the network when failures occur.

The main configuration parameter for channel persistence is the Link Class for each interface. The Link Class decides how a node shall behave if it detects that the neighboring node stops responding or the link fails. It is configured on a per-interface basis. The link class must be configured with the same value at both ends of a link.

Link Class Normal

Link Class Normal means that the node will consider the link as *Down* if it detects an error with the link (Signal Failure) or if the node at the other end of the link fails to respond to the periodic supervision messages. When a link is considered as *Down*, all channels utilizing that link will be immediately torn down. This means that as soon as an error is detected, all channels that are affected by the error are torn down. Link Class *Normal* is the default for all interfaces.

An interface configured with link class *Normal* will never have status *NoControl* or *DownKeep*. It will go directly to status *Down* instead

Link Class Persistent

With Link Class *Persistent*, the node will not tear down channels if the neighboring node stops responding. Instead, it will classify the link as *NoControl* and leave all existing channels in place, but deny any new channels from being established via the interface to the peering node that has stopped responding. Furthermore, if a node has one or more links configured as *Persistent*, the node will by default enter a *NoControl* mode if the node-controller restarts. In the *NoControl* mode, the node will not run the normal DTM protocol stack and instead leave the hardware configured as-is. This means that all existing channels will continue to forward data, but it will not be possible to supervise the links and channels or setup any new channels.

If an interface is configured with Link Class *Persistent* and the node receives an indication that the link on that interface has failed completely (e.g. a Signal Failure condition), it will tear down all channels over that link.

© 2003-2010 Net Insight AB, All Rights Reserved

Persistent links are useful to protect end-nodes that are single points of failure for a service.

An interface configured with link-class *Persistent* will never have status *DownKeep*, it will go to status *Down* instead.

Link Class Nailed

With Link Class *Nailed*, the node will never tear down channels unless the operator sets the administrative status of the interface to down or if the channel is removed via the management interface. This means that the node will leave channels in place even if it knows that the channels cannot forward any data since there is a loss-of-signal situation on the interface.

If the neighboring node stops responding or if the link to or from the neighboring node fails, the node will leave all existing channels in place, but deny any new channels from being established via the interface to the peering node. Furthermore, if a node has one or more links configured as *Nailed*, the node will by default enter a *NoControl* mode if the node-controller restarts. In the *NoControl* mode, the node will not run the normal DTM protocol stack and instead leave the hardware configured as-is. This means that all existing channels will continue to forward data, but it will not be possible to supervise the links and channels or setup any new channels.

Nailed links are useful since they allow links that break in one direction to continue forwarding data in the other direction. They also allow faster recovery after the link has been restored again since all channels are already established.

See Also

Element Manager User's Manual for the Nimbra (document NID601).

Trace Network Synchronization

Trace synchronization is a function that traces how the synchronization is distributed in the network. If Time Transfer is available, its distribution will also be traced.

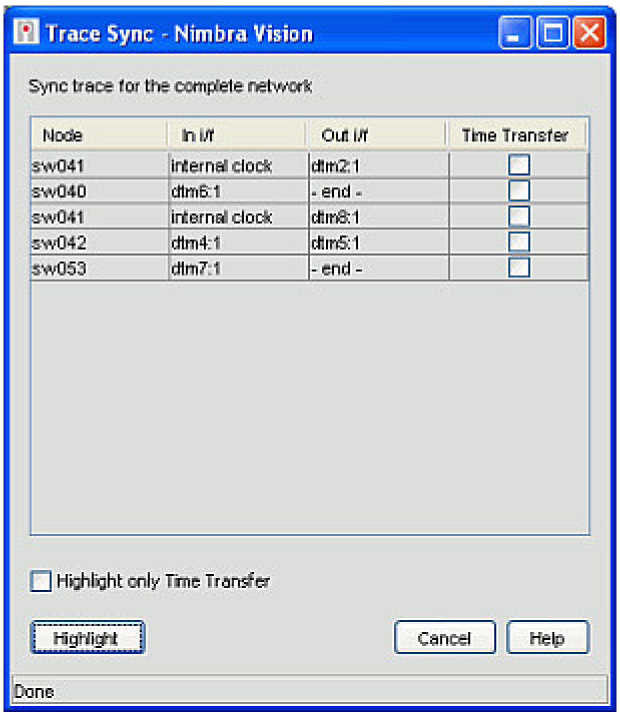
Trace the Synchronization

To trace the synchronization distribution through the network, select a Nimbra node in a map or the network database, and select menu **Nimbra Node | Trace Sync…**. This opens a dialog where you have the following choices:

- **Trace sync from sub network root**. This back-traces the sync distribution from the selected node to the sync root. I.e., it traces how the sync is distributed from the sync root to the selected node.
- **Trace complete sub network**. This traces the sync distribution from the sync root of the selected node. I.e., it traces how the sync is distributed from the sync root of the selected node to all the nodes. You may have selected any node that is getting its sync from the same sync root, and should get the same result.
- **Trace complete network**. This traces the sync distribution to all the nodes, i.e. also nodes having other sync roots that the one of the selected node. You may have selected any node, and should get the same result.

The result of the sync trace is displayed as an ordered list of next-hop nodes, starting with the sync root.

© 2003-2010 Net Insight AB, All Rights Reserved



Sync trace for the complete network

Node	In i/f	Out i/f	Time Transfer
sw041	internal clock	dtm2:1	☐
sw040	dtm6:1	- end -	☐
sw041	internal clock	dtm8:1	☐
sw042	dtm4:1	dtm5:1	☐
sw053	dtm7:1	- end -	☐

☐ Highlight only Time Transfer

Done

Sync trace result window.

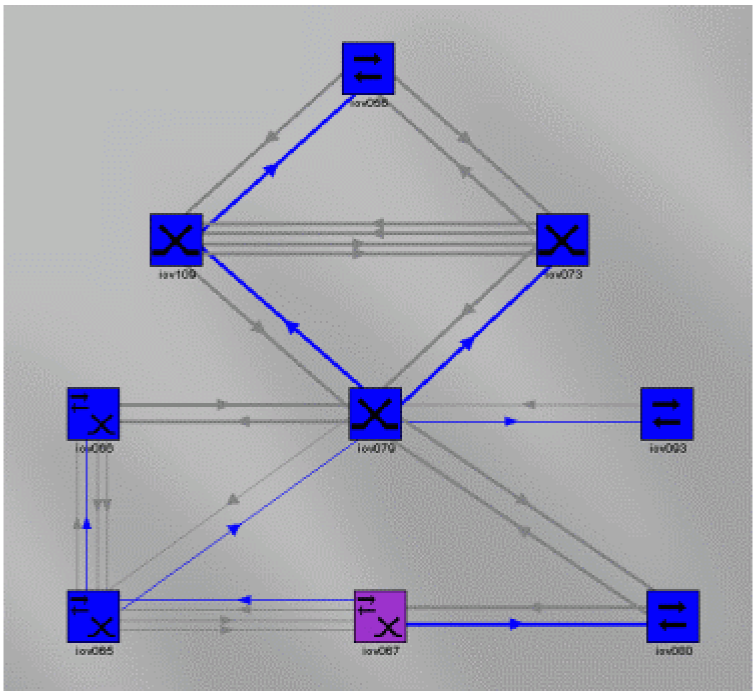
- **Node**. The name of the node.
- **In i/f**. The name of the DTM interface where the sync is received from the previous node. The sync root is not receiving a sync signal from its DTM interface. In that case, this is the name of the external sync input interface, or the local oscillator.
- **Out i/f**. The name of the DTM interface where the sync is distributed to the next node. If the node does not distribute the sync, this is indicated with "- end -".
- **Time Transfer**. Indicates if the node is using Time Transfer.

Display Result in Map

The traced sync path can be visually displayed in all maps as highlighted symbols. To highlight the path in the maps, select the button **Highlight**. When Time Transfer is highlighted, then the nodes using Time Transfer is highlighted as well as the links originating in these nodes. The following colors are used when highlighting sync trace:

Purpose	Color Name	Color
Sync root	Dark Orchid	🟪
Sync distribution, no Time Transfer	Blue	🟦
Sync distribution, with Time Transfer	Deep Sky Blue	🟦

© 2003-2010 Net Insight AB, All Rights Reserved



Detail of map with a highlighted sync trace.

How Sync Trace Works

The synchronization information of the nodes is retrieved at the periodic status polling and stored in the database with the Nimbra node managed object. This is normally done every 2 minutes. When a sync trace is started, the information in the database, together with the selected node and the desired function, is used.

See Also

Highlight in Maps

Redundant Headends

Redundant Headends

Function

This chapter describes the redundant headend function. The function monitors a service (primary), and in case of failure on the service, it activates a backup service (secondary).

When using a redundant headend, two services must be configured in the Nimbra nodes: a primary and a secondary service. The primary service shall be enabled (administrative status *up*), and the secondary shall be disabled (administrative status *down*). If the primary service somehow fails, the function enables the secondary service and disables the primary service. If the primary service is disabled (i.e. its administrative status is *down*) then the redundant headend function for this service is disabled and it is not monitored.

The function monitors the primary access interface and service status of the headend node. This allows for redundancy on the input data stream, the ingress port and the ingress board. The function also monitors the trunk interfaces on all neighboring nodes that terminate trunks from the primary node. This allows for node redundancy.

If a fault is detected on the monitored access interface or on the service, the function will disable the primary service. This will free network resources and

© 2003-2010 Net Insight AB, All Rights Reserved

allow the secondary service to connect. After disabling the primary service, it will enable the secondary service, and the service is thus switched over.

If a fault is detected on all supervised trunk interfaces, the function will assume that the node with the primary service is down, and will enable the secondary service. (A periodic status poll would also detect if a node is unreachable, but this is faster.) Because the node is most likely down, the function will at this stage not try to disable the primary service. Instead, the primary service is disabled when one trunk is detected to be up again, i.e. when the node is likely to be reachable again.

The function makes its decision based on alarms that Nimbra Vision receives from the nodes. It is therefore important that the trap notification receiver is correctly configured on the nodes. Note that the *Loss of Signal* alarms must not be suppressed on the supervised trunk interfaces.

Monitored entity	**Alarm**	**Cause**
Access interface	Loss of Signal	Indicates an error in the input signal.
Access interface	Loss of Frame	
Access Interface	Threshold Crossed (Frequency out of range)	
TTP (service)	Underlying resource unavailable	Indicates a broken or absent interface board.
Trunk interface	Loss of Signal	Indicates lost signal from neighboring node. If detected on all neighboring nodes, then headend node is assumed down.
Trunk interface	Loss of Frame	
Trunk interface	Alarm Indication Signal	

The function can be used for different types of redundant headends:

- Redundant access interfaces/boards. Feeding the same signal into two interfaces on the same node. The secondary service (TTP) is configured on the same node as the primary service. In case of failure on one of the feeds, or failure on the access interface, the function switches to the secondary service.
- Redundant nodes. Feeding the same signal into two different nodes.
- Alternative service. Feeding the same signal into two different service, where the services terminate on different destinations.

Automatic Enabling/Disabling of Access Interfaces

For ITS services, the operational state of the access interface is always following the operational state of the corresponding service (TTP). This state is always controlled by the node, and is not controlled by the redundant headends function.

For ETS service, the relationship between the access interface (Ethernet port) and the service (TTP) is not so strict. The Ethernet interface may e.g. be used by multiple services (TTPs). The administrative state is therefore not automatically controlled by the service itself. The redundant headends function can therefore optionally control the state of the Ethernet interfaces. If this option is enabled, then when the function makes a switch-over, it disables all Ethernet interfaces on the primary service, and enables all Ethernet interfaces on the secondary service. The function disables all Ethernet interfaces on the primary node associated with the primary service, and all Ethernet interfaces associated with all destinations.

© 2003-2010 Net Insight AB, All Rights Reserved

Similarly, it enables the corresponding secondary interfaces. The function controls all interfaces associated with the originating TTP, and all interfaces associated with the terminating TTP. If the service is a multicast service, then all destinations are affected.

Logging

When the filter enables and disables services (TTP's), this is logged by the Configuration Server. The log can be accessed from the Java Client, at the tree node **Configuration | Audit**.

See Also

Configure Redundant Headends

> ***Note!*** *An alarm is raised if attempting to use the function without required license. The alarm is cleared when the required license is installed.*

Configure Redundant Headends

To configure and edit redundant headends, select the node **Configure | Redundant Headends** from the tree. This opens a panel with a table containing all configured redundant headends. Some menus and toolbar buttons are associated with the panel.

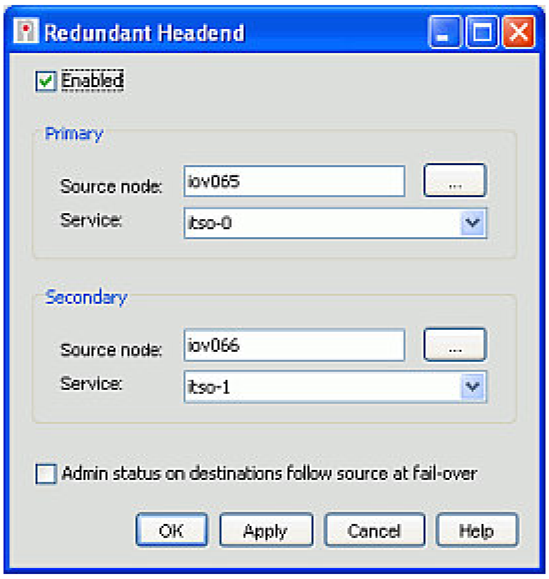
To add a new redundant headend object, use menu **Edit | Add**. This creates an object and opens a dialog to edit the new object.

To copy an existing redundant headend object, select a row in the table and use menu **Edit | Copy**. This copies the selected row and opens a dialog to edit the new object.

To modify an existing redundant headend object, select a row and use menu **Edit | Edit**. This opens a dialog to edit the selected object.

To delete a redundant headend object, select the row(s) and use menu **Edit | Delete**. This deleted the selected objects(s).

To enable or disable a redundant headend objects, select the row(s) and use menus **Action | Enable** or **Action | Disable**.


Redundant Headend

[✓] Enabled

Primary

Source node: iov065

Service: itso-0

Secondary

Source node: iov066

Service: itso-1

[ ] Admin status on destinations follow source at fail-over

Dialog to add or edit redundant headend configurations.

Enable. Check the checkbox to enable the redundant headend.

The primary service is the service that is monitored:

© 2003-2010 Net Insight AB, All Rights Reserved

- **Source node**. The name of the node where the primary service originates.
- **Service**. The name of the primary service.

The secondary service is enabled in case of failure on the primary service:

- **Source node**. The name of the node where the secondary service originates.
- **Service**. The name of the secondary service.

If the checkbox **Admin status on destinations follow source at fail-over** is checked, then all Ethernet interfaces will be disabled and enabled according to the status of the primary and secondary service. This is only applicable for ETS. See Redundant Headends for details.

Note! *The function does not switch back from the secondary service to the primary service when the fault condition clears.*

Tip! *You can configure two redundant headends to protect each other. Configure one redundant headend where A is protected and B is secondary, and another where B is protected and A is secondary.*

Note! *If you have two redundant headends protecting each other, and both of them has faults, then the function will toggle back and forth between the two services.*

Note! *To disable the redundant headend for a service, you can set its administrative state to* down. *This means that the administrative state must be* up *and the redundant headend must be enabled for the function to be enabled.*

All switch-over that is done by the function is logged in the Audit Log, which can be accessed from the Tree node **Configuration | Audit**.

See Also

Redundant Headends

Pre-emption

About Pre-emption

In network failure situations it is important that sufficient link bandwidth is available for high-priority services to be rerouted. Using the pre-emption function, Nimbra Vision will allow prioritized services to be rerouted by pre-empting low-priority services, thereby making the necessary bandwidth available.

The user defines the links, high-priority and low-priority services subject to pre-emption in Nimbra Vision. Upon failure of any of the links, Nimbra Vision will pre-empt all low-priority services. Nimbra Vision will also pre-empt the low-priority services if any of the high-priority services fails to establish a connection due to congestion (i.e. lack of bandwidth). Pre-emption may either be defined as disabling of the service completely or as reduction of service bandwidth.

A pre-emption object represents the configuration of one pre-emption function, with its links and services. A pre-emption object can be in state *Full Operation* or *Reduced Operation*. When in *Full Operation*, no services are pre-empted. When in *Reduced Operation*, all the low-priority (reduced) services are pre-empted. The status changes depending on the status of the monitored links and monitored high-priority services as described below.

A pre-emption object consists of a set of monitored DTM interfaces that represents the monitored links, a set of monitored services that represents the

© 2003-2010 Net Insight AB, All Rights Reserved

high-priority services, and a set of reduced services representing the low-priority services. Each reduced service has a capacity to use when the pre-emption object is in *Full Operation* and a capacity to use when in *Reduced Operation*. Whenever the pre-emption object changes its status between *Full Operation* and *Reduced Operation*, the capacity on the reduced services are changed according to the configured full operation or reduced operation capacity.

If the pre-emption object detects a congestion alarm on any of its monitored services, then its status is changed to *Reduced Operation*. A congestion alarm describes that the service fails to establish its connection because of lack of bandwidth to the destination. When services are now being pre-empted, then the monitored service should be able to successfully establish its connection, and its alarm is cleared. If a monitored service is now being disabled (by setting its administrative status to down), then the pre-emption object will detect this and attempted to restore its status to *Full Operation* (unless it should remain in *Reduced Operation* because of the status on monitored DTM interfaces, as being described below). If at this time any of the reduced services fail to re-establish its connections because of a lack of bandwidth, then the status is returned to *Reduced Operation*.

In NimOS prior to GX4.6.0.0, congestion alarms are not generated for multicast services. A monitored service should therefore not be a multicast service unless NimOS GX4.6.0.0 or later is used.

If the pre-emption object detects an alarm with cause *underlyingResourceUnavailable* on any of the monitored DTM interfaces, then its status is changed to *Reduced Operation*. This alarm is raised when the trunk interface fails to terminate a link. The pre-emption object will not restore the status to *Full Operation* while any of the monitored DTM interfaces have this alarm. When all the alarms are cleared, its status returns to *Full Operation* (unless it should remain *in Reduced Operation* because of status on monitored service as described earlier).

It is possible to set up multiple instances of pre-emption, i.e. multiple pre-emption objects, for different parts of the network. It is also possible to include a service in more than one pre-emption object. In this case, if multiple pre-emption objects have status *Reduced Operation*, then the lowest configured reduced capacity is set on the reduced service.

State Transitions Summary

Full Operation to *Reduced Operation*, when any of:

- Alarm *underlyingResourceUnavaiable* on any the monitored DTM interfaces.
- Has monitored service that is enabled and receives alarm *callEstablishmentError* or *congestion* that indicates congestion on any monitored or reduced service.

Reduced Operation to *Full Operation*, when all of:

- No *underlyingResourceUnavailable* alarm on any of the monitored DTM interfaces.
- No monitored service is enabled, or has not detected *callEstablishmentError* or *congestion* that indicates congestion on any of the monitored or reduced services.

© 2003-2010 Net Insight AB, All Rights Reserved

Example 1

You have a network with redundant paths, A and B. Path A does not have enough capacity for all the services, so during normal operation services are running over both path A and path B. Whenever path B is unavailable, then a low-priority ETS services should have its capacity reduced from 300 to 20 Mbps, and a low-priority ASI of 45 Mbps service should be disabled. This would release enough capacity for the other services to re-establish using only path A. To solve this using pre-emption, create a pre-emption object. Add the terminating DTM interfaces for all the links in path B as monitored DTM interfaces. Add the low-priority ETS service to the reduced services, with its full capacity set to 300 Mbps, and its reduced capacity set to 20 Mbps. Add the low-priority ASI service to the reduced services, with its full capacity set to 45 Mbps, and its reduced capacity set to 0 Mbps (this means disabled). Now, if a link goes down in path B, an alarm is raised on its terminating DTM interface and the pre-emption object changes status to *Reduced Operation*. All the services that were using path B has been torn down and are being re-established by the network. When the pre-emption object has changed the capacity on the ETS service to 20 Mbps, and disabled the ASI connection, then the network will be able to re-establish the remaining connections using only path A.

Later, when path B is being available again, the DTM interface clears its alarm. The pre-emption object detects this, and restores its status to *Full Operation*. The ETS connection is now being re-established with its full operation capacity 300 Mbps and the ASI connection is being established with its full operation capacity 45 Mbps.

Example 2

You have a number of services sharing some part of the network. One of these services is an occasional-use SDI service with high priority. You also have a best-effort ETS service. When the occasional-use service is being used, then the ETS service can use 100 Mbps. When the occasional-use service is not used, you want to use the network capacity otherwise used by the occasional use SDI service for the ETS service. To solve this using pre-emption, create a pre-emption object. Add the occasional-use SDI service as a monitored service. No specific links are to be monitored, so nothing is added to the monitored DTM interfaces. Add the best-effort ETS service to the reduced services. Set its full operation capacity to 370 Mbps, and its reduced capacity to 100 Mbps. Now, whenever the occasional-use SDI service is being used, it will initially fail because of a lack of bandwidth. The pre-emption object changes state to *Reduced Operation* and the ETS service is being re-established with the reduced capacity 100 Mbps. We haven now freed 270 Mbps in the network, and the occasional-use SDI service is now able to successfully establish its connection, and the alarm is cleared.

Later, when you no longer are using the occasional-use SDI service, you set its administrative state to down. The pre-emption function detects this, and restores its state to full operation. The ETS service is then being re-established with its full operation capacity 370 Mbps.

See Also

Configure Pre-emption

© 2003-2010 Net Insight AB, All Rights Reserved

Configure Pre-emption

To configure and edit pre-emption, select the node **Configure | Preemption** from the tree. This opens a panel with a table containing all configured pre-emption objects. Some menus and toolbar buttons are associated with the panel.

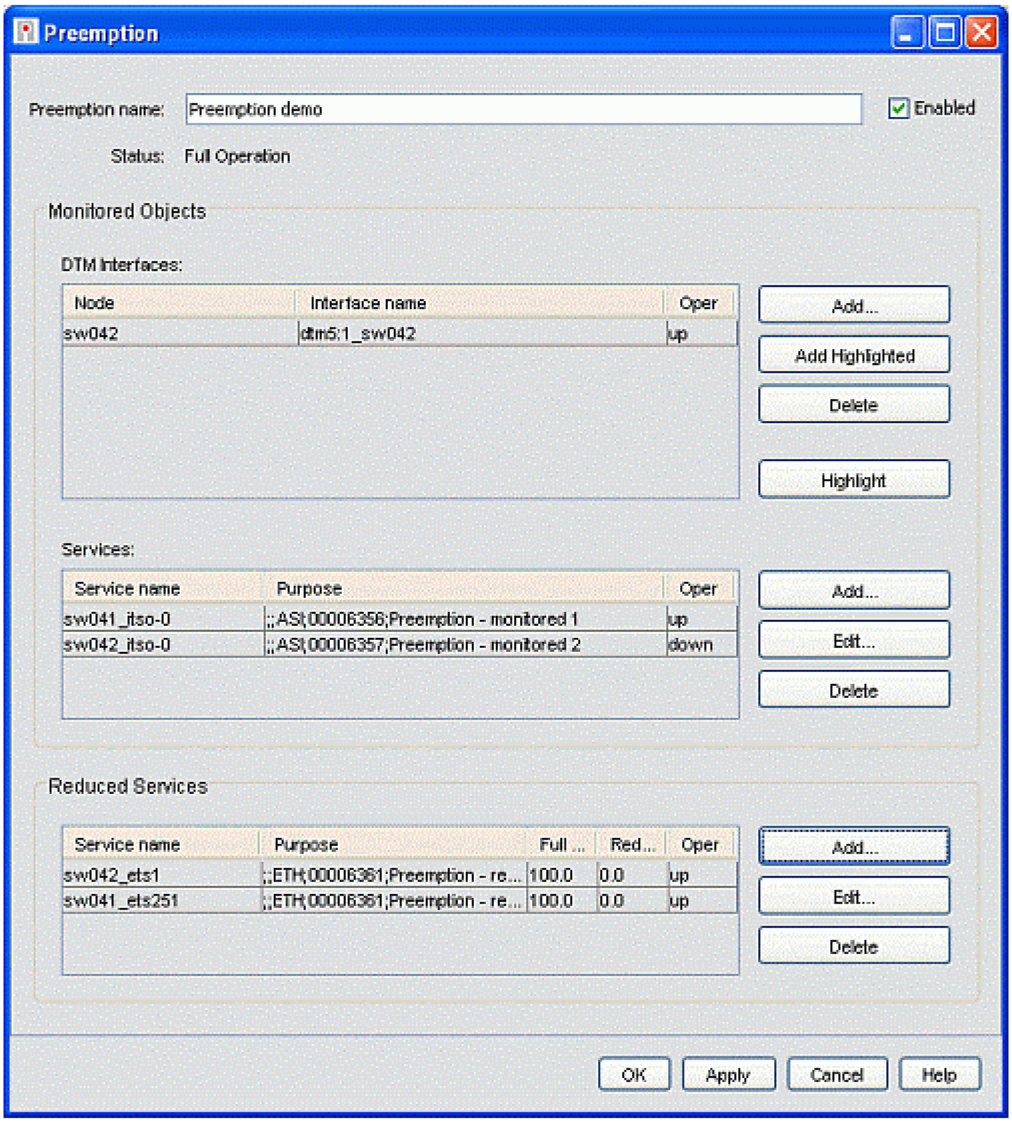
To add a new pre-emption object, use menu **Edit | Add**. This creates an object and opens a dialog to edit the new object.

To copy an existing pre-emption object, select a row in the table and use menu **Edit | Copy**. This copies the selected row and opens a dialog to edit the new object.

To modify an existing pre-emption object, select a row and use menu **Edit | Edit**. This opens a dialog to edit the selected object.

To delete a pre-emption object, select the row(s) and use menu **Edit | Delete**. This deleted the selected objects(s).

To enable or disable pre-emption object, select the row(s) and use menus **Action | Enable** or **Action | Disable**.



Preemption name: Preemption demo ✓ Enabled

Status: Full Operation

Monitored Objects

DTM Interfaces:

Node	Interface name	Oper
sw042	dtm5:1_sw042	up

Services:

Service name	Purpose	Oper
sw041_itso-0	;;ASI;00006356;Preemption - monitored 1	up
sw042_itso-0	;;ASI;00006357;Preemption - monitored 2	down

Reduced Services

Service name	Purpose	Full ...	Red...	Oper
sw042_ets1	;;ETH;00006361;Preemption - re...	100.0	0.0	up
sw041_ets251	;;ETH;00006361;Preemption - re...	100.0	0.0	up

Edit pre-emption configuration details dialog.

- **Preemption name**: a name of the pre-emption object. This name will appear in the **Configure | Preemption**. You can use any name.

© 2003-2010 Net Insight AB, All Rights Reserved

- **Enabled**. If this checkbox is checked, then this pre-emption object will be active. It will monitor the objects and change its status accordingly.
- **Status**. The status is either *Full Operation* or *Reduced Operation*.
- **Monitored DTM Interfaces**. This is a table of DTM interfaces that shall be monitored. If a DTM interface in this table gets an alarm with cause *underlyingResourceUnavailabe*, then the pre-emption object changes its state to *Reduced Operation*.
    - The column **Node** is the name of the node where the interface is located.
    - The column **Interface name** is the name of the interface managed object.
    - The column **Oper** is the operational status of the interface.
    - Click **Add…** to add an interface to the table. This opens a search dialog.
    - Click **Add Highlight** to add the terminating interfaces from all links highlighted in maps.
    - Click **Delete** to delete the selected row from the table.
    - Click **Highlight** to highlight all links that terminate in interfaces in the table.
- **Monitored Services**. This is a table of service to monitor. If a service fails to establish its connection because of a lack of bandwidth, then the pre-emption object changes status to *Reduced Operation*.
    - The column **Service name** is the name of the service
    - The column **Purpose** is the purpose of the service.
    - The column **Oper** is the operational state of the service.
    - Click **Add…** to add a service to the table. This opens a search dialog.
    - Click **Edit…** to change the selected service entry, i.e. replace it with another.
    - Click **Delete** to delete the selected row from the table.
- **Reduced services**. This is a table of services and their capacity setting when the pre-emption object is in *Full Operation* or *Reduced Operation*. The capacity shall be set in Mbps (or MHz if applicable, e.g. for AES/EBU). If the capacity is set to 0, then instead of changing the capacity, the administrative state is set to *up* (*Full Operation*) or *down* (*Reduced Operation*).
    - The column **Service name** is the name of the service.
    - The column **Purpose** is the purpose of the service.
    - The column **Full Capacity** is the capacity to set on a service when the pre-emption object is changing its status to *Full Operation*.
    - The column **Reduced Capacity** is the capacity to set on a service when the pre-emption object is changing its status to *Reduced Operation*. If this capacity is set to 0, then the administrative state of the service is instead set to *down*.
    - The column **Oper** is the operational state of the service.
    - Click **Add…** to add a service to the table. This opens a dialog to search for a service, and to configure its capacities.

© 2003-2010 Net Insight AB, All Rights Reserved

- Click **Edit…** to edit the selected service entry, including the capacity settings.
- Click **Delete** to delete the row from the table.

See Also

About Pre-emption

Network Inventory

About Inventory

Nimbra Vision supports centralized inventory of all network elements in a Nimbra network. Inventory data is automatically discovered and stored in the Nimbra Vision database. The data includes all Field Replaceable Units (FRU) from all network elements, such as hardware modules, application packages, software, etc.

Inventory data is searchable with the result presented in a sortable table. Using this search function it is possible for instance to find out where a certain type of FRU is installed. It is possible to configure what information (columns) to be displayed in the resulting table. Inventory objects and their relationship may also be browsed per network element in a tree structure, where nodes may be expanded and collapsed

Inventory data can also be exported to files.

The following topics describe the function in more details:

- Search/Browse Inventory Database
- Inventory Search Result
- Inventory Properties
- Export Inventory Policy

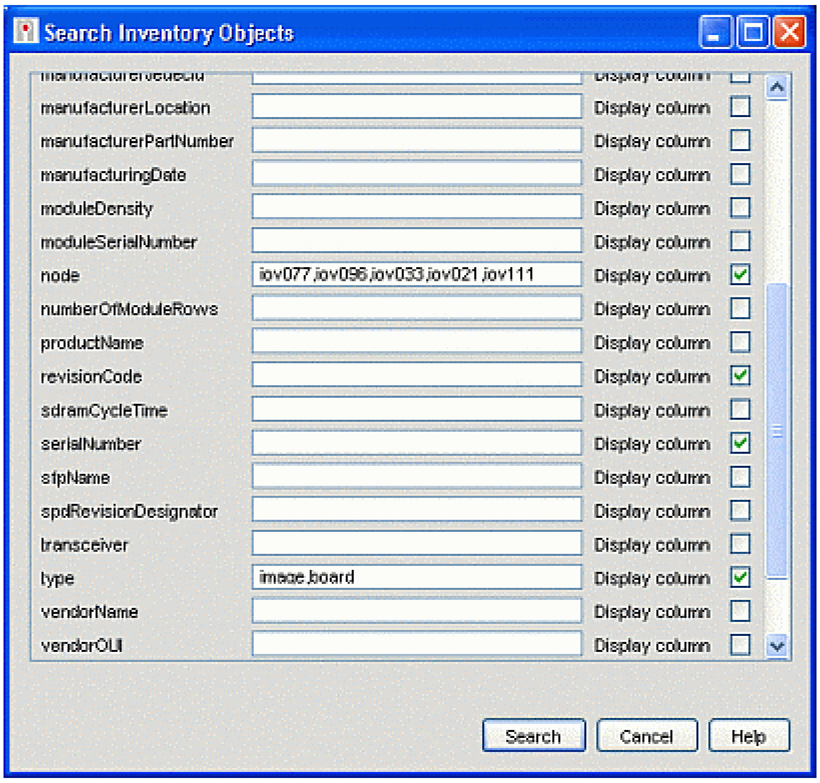
Search/Browse Inventory Database

To search the inventory database, choose menu **View | Inventory Search…** . With this function, you can enter criteria for any inventory property. The result is presented in a table.

To browse the inventory database in a tree view, choose menu **View | Inventory Browse…** . With this function, you can browse the inventory tree for some nodes. The only criterion available is the node. The result can also be presented in a table.

If nodes are selected in the map or database when starting the search/browse, these will be added to the node search criterion.

© 2003-2010 Net Insight AB, All Rights Reserved



manufacturerLocation		Display column	☐
manufacturerPartNumber		Display column	☐
manufacturingDate		Display column	☐
moduleDensity		Display column	☐
moduleSerialNumber		Display column	☐
node	iov077,iov096,iov033,iov021,iov111	Display column	✓
numberOfModuleRows		Display column	☐
productName		Display column	☐
revisionCode		Display column	✓
sdramCycleTime		Display column	☐
serialNumber		Display column	✓
sfpName		Display column	☐
spdRevisionDesignator		Display column	☐
transceiver		Display column	☐
type	image,board	Display column	✓
vendorName		Display column	☐
vendorOUI		Display column	☐

Searching or browsing the inventory database.

Enter search criteria for each property. Only inventory objects matching the criteria is displayed as result of the search. See Inventory Properties for description on the different properties. See Writing Filer Criteria on how to write criteria.

Display column: if ticked, then the column is displayed in the tabular search result.

Click **Search** to search and present the result.

***Note!** You must have a Node user defined for the network elements. See Node Users in the Administrator's Guide.*

See Also

Inventory Search Result

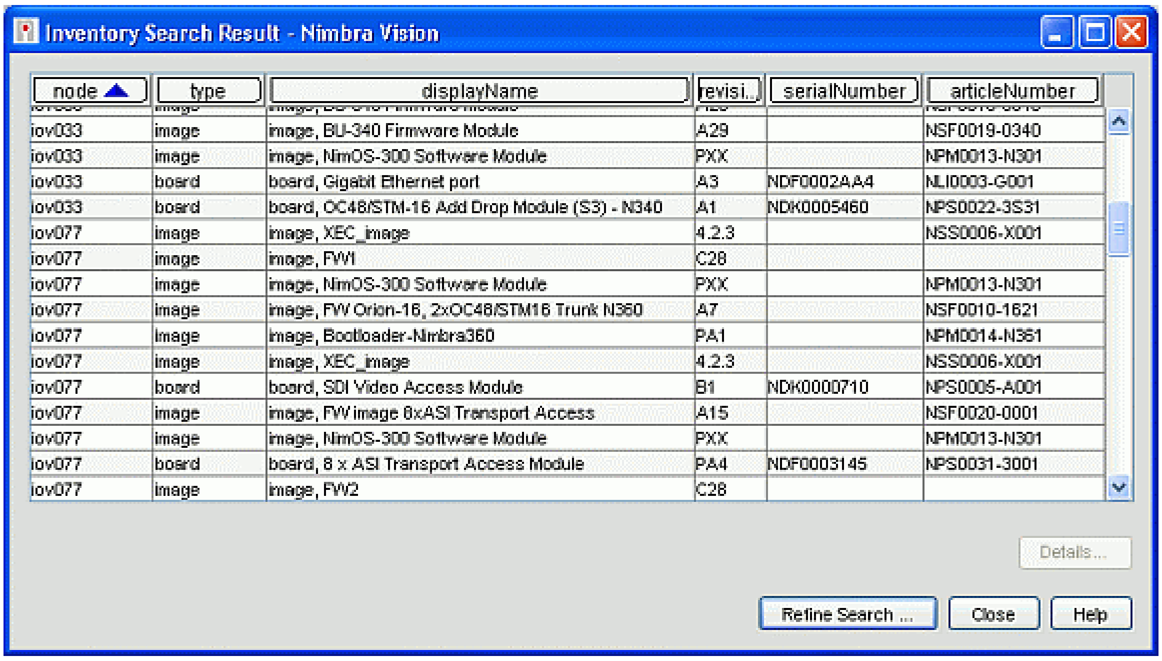
Inventory Properties

Writing Filter Criteria

Inventory Search Result

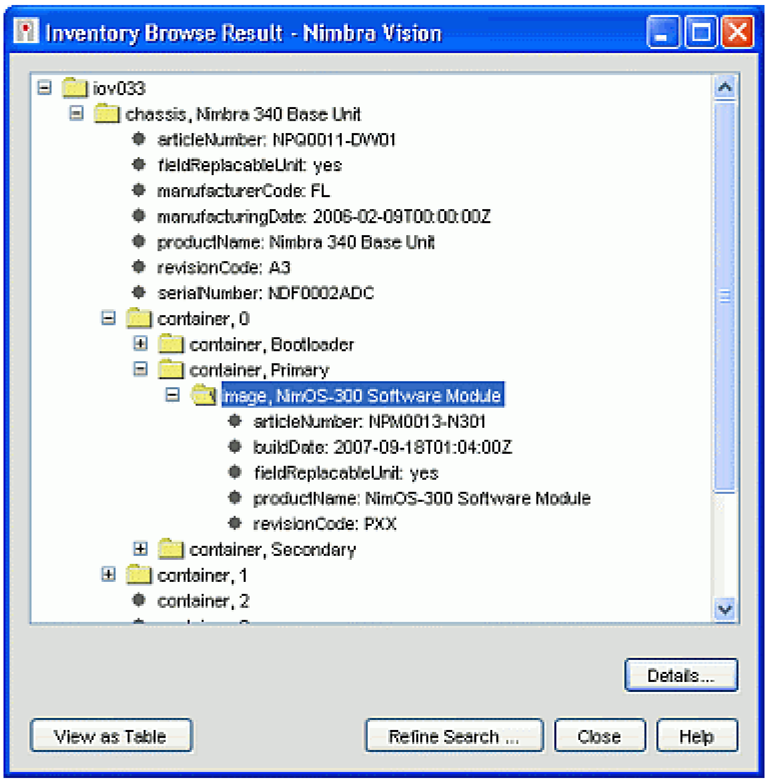
The inventory search result dialog displays the inventory objects, either as a list or as a tree.

© 2003-2010 Net Insight AB, All Rights Reserved


Inventory Search Result - Nimbra Vision

node	type	displayName	revisi..	serialNumber	articleNumber
iov033	image	image, BU-340 Firmware Module	A29		NSF0019-0340
iov033	image	image, NimOS-300 Software Module	PXX		NPM0013-N301
iov033	board	board, Gigabit Ethernet port	A3	NDF0002AA4	NLI0003-G001
iov033	board	board, OC48/STM-16 Add Drop Module (S3) - N340	A1	NDK0005460	NPS0022-3S31
iov077	image	image, XEC_image	4.2.3		NSS0006-X001
iov077	image	image, FW1	C28		
iov077	image	image, NimOS-300 Software Module	PXX		NPM0013-N301
iov077	image	image, FW Orion-16, 2xOC48/STM16 Trunk N360	A7		NSF0010-1621
iov077	image	image, Bootloader-Nimbra360	PA1		NPM0014-N361
iov077	image	image, XEC_image	4.2.3		NSS0006-X001
iov077	board	board, SDI Video Access Module	B1	NDK0000710	NPS0005-A001
iov077	image	image, FW image 8xASI Transport Access	A15		NSF0020-0001
iov077	image	image, NimOS-300 Software Module	PXX		NPM0013-N301
iov077	board	board, 8 x ASI Transport Access Module	PA4	NDF0003145	NPS0031-3001
iov077	image	image, FW2	C28		

Result from Inventory Search.


Inventory Browse Result - Nimbra Vision

```
⊟ iov033
  ⊟ chassis, Nimbra 340 Base Unit
      ● articleNumber: NPQ0011-DW01
      ● fieldReplacableUnit: yes
      ● manufacturerCode: FL
      ● manufacturingDate: 2006-02-09T00:00:00Z
      ● productName: Nimbra 340 Base Unit
      ● revisionCode: A3
      ● serialNumber: NDF0002ADC
    ⊟ container, 0
      ⊞ container, Bootloader
      ⊟ container, Primary
        ⊟ image, NimOS-300 Software Module
            ● articleNumber: NPM0013-N301
            ● buildDate: 2007-09-18T01:04:00Z
            ● fieldReplacableUnit: yes
            ● productName: NimOS-300 Software Module
            ● revisionCode: PXX
      ⊞ container, Secondary
    ⊞ container, 1
      ● container, 2
```

Inventory browser tree.

Click **Details…** to view all the properties for the selected inventory object.

Click **Refine Search…** to open the search dialog again.

Click **View as Table** to display the inventory objects in a tabular format instead of tree format.

Click **View as Tree** to display the inventory objects in a tree instead of a tabular format.

Click **Close** to exit the function.

© 2003-2010 Net Insight AB, All Rights Reserved

See Also

Search/Browse Inventory Database

Inventory Properties

Inventory Properties

Propery	Applies to	Description
articleNumber		Net Insight article number of the entity.
bitRate	sfp	SFP bit rate in Mbps.
buildDate	image	Time stamp when the image was built, on format ISO-8601.
connectorType	sfp	
container		The name (containerName) of the container where the entity is mounted. This is the name of the parent entity if the entity is mounted in a container.
containerName	container	The name of the container, e.g. `Primary`, `Secondary`, `Running`, `IF 1`. A container is a location where a typically a field replaceable unit is mounted or installed.
dateCode	sfp	Date as specified by vendor.
displayName	*(all)*	A somewhat more descriptive name of the entity. Nimbra Vision constructs this property only upon display in Search of Browse Inventory. It is not displayed in Export Inventory policy. It is not possible to search in database on display name.
fieldReplacableUnit		Boolean value describing if the entity is considered a FRU (Filed Replicable Unit) or not. Any of `true` of `false`.
fundamentalMemoryType	ram	
length	sfp	Length. If the SFP supports multiple lengths, then Multiple is presented.
manufacturerCode		Code representing the manufacturer.
manufacturerJedecId	ram	
manufacturerLocation	ram	Code representing the location of the manufacturer.
manufacturerPartNumber	ram	Part number (article number) as specified by the vendor.
manufacturingDate		Time stamp when the entity was manufactured, on format ISO-8601.
moduleDensity	ram	
moduleSerialNumber	ram	Serial number.

© 2003-2010 Net Insight AB, All Rights Reserved

Propery	Applies to	Description
node		The name of the network element in Nimbra Vision database.
numberOfModuleRows	ram	
productName		The Net Insight name of the entity, as reported by the entity.
revisionCode		The revision code.
sdramCycleTime	ram	
serialNumber		Net Insight serial number.
sfpName	sfp	A constructed name of the SFP as reported by the network element. An SFP does not have a programmed name. Instead, a name is constructed.
spdRevisionDesignator	ram	
transceiver	sfp	As specified by the vendor.
type		Describes the type of entity. Any of: • chassis • board • container • fan • image • power • ram • sfp
vendorName	sfp	The name of the vendor.
vendorOUI	sfp	As specified by the vendor.
vendorPartNumber	sfp	Article number (part number) as specified by the vendor.
vendorRevision	sfp	Revision as specified by the vendor.
vendorSerialNumber	sfp	Serial number as specified by the vendor.

Software and Firmware Upload

About Software/Firmware Upload

It is possible to download a complete software and firmware package to multiple specified network elements in a single command. The network elements are selected from the Nimbra Vision map or the network database. The software to be downloaded is stored in a repository, which is given a name and a URL describing its location. Multiple repositories may be defined for different software and firmware distributions. A distribution typically contains all the software and firmware for all types of Nimbra network elements.

When software and firmware upload is complete, the nodes may either be restarted automatically or manually restarted at a convenient time.

The following topics described the function in more detail:

© 2003-2010 Net Insight AB, All Rights Reserved

- Installing the Repository
- Manage Software/Firmware Upload Repositories
- Uploading Software/Firmware
- Functional Description of Software/Firmware Upload
- Configure Software/Firmware Upload

Installing the Repository

The software and firmware distribution is delivered as a single file. This is as compressed tar file containing all the images, typically for all the type of Nimbra network elements.

The Software/Firmware Upload function is loading the software and firmware images from a repository using HTTP or anonymous FTP.

Prior uploading, you must install the repository on a HTTP server or server FTP. The network elements must have access to this HTTP or FTP server. To install, unpack and extract the compressed tar file on the FTP server or HTTP server. This will create a folder with the same name as the system release, e.g. `GX4.3.0.0`.

You can use the Nimbra Vision server as a repository server. To do that:

1. Create a sub-folder in the Nimbra Vision home folder (e.g. *<NMS_Home>*`\repos`). This folder will hold all the repositories.
2. Extract the single file repository to the folder. This creates a sub-folder that contains all the images. You will need to get a file uncompress software do this. Windows does include software for extracting tar files.
3. Done. The URL to the repository is then `http://`*server*`:9090/repos/`*RELEASE*.

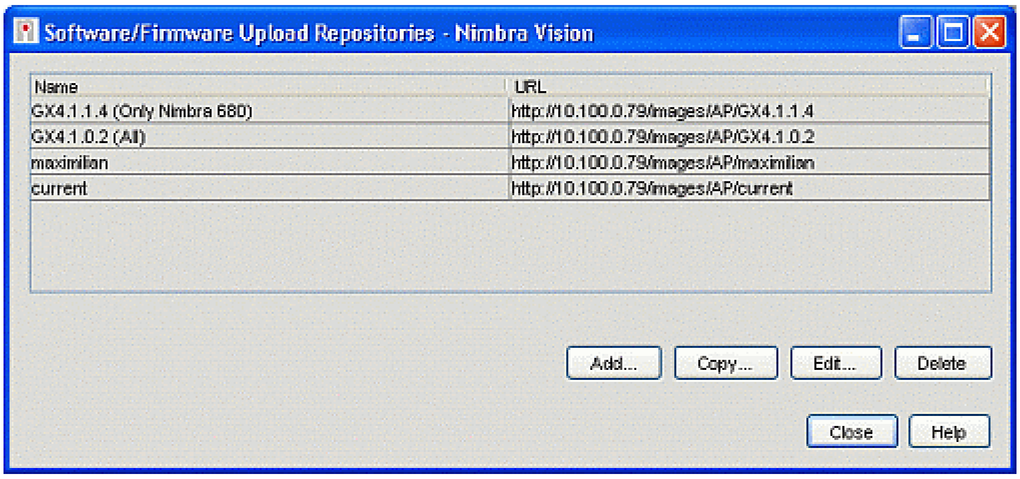
Manage Software/Firmware Upload Repositories

This dialog allows you to define repositories. A software/firmware repository contains all software and firmware images for a release, a complete software and firmware distribution. The repositories are given a name and an URL describing its location.

To edit definitions of repositories, choose menu **Tools | Manage Software/Firmware Upload Repositories…**. This opens the dialog to add, edit and delete definitions of repositories.

The dialog contains a list if repositories, where it associates a repository name with a repository location (URL). The repository names will appear in the Upload Software/Firmware dialog.

© 2003-2010 Net Insight AB, All Rights Reserved



Name	URL
GX4.1.1.4 (Only Nimbra 680)	http://10.100.0.79/images/AP/GX4.1.1.4
GX4.1.0.2 (All)	http://10.100.0.79/images/AP/GX4.1.0.2
maximilian	http://10.100.0.79/images/AP/maximilian
current	http://10.100.0.79/images/AP/current

Manage Software/Firmware Upload Repositories.

Click **Add…** to define a new repository.

Click **Edit…** to edit the definition of an existing repository.

Click **Copy…** to copy the definition of an existing repository.

Click **Delete** to delete the definition of a repository.

Click **Close** to exit the window.

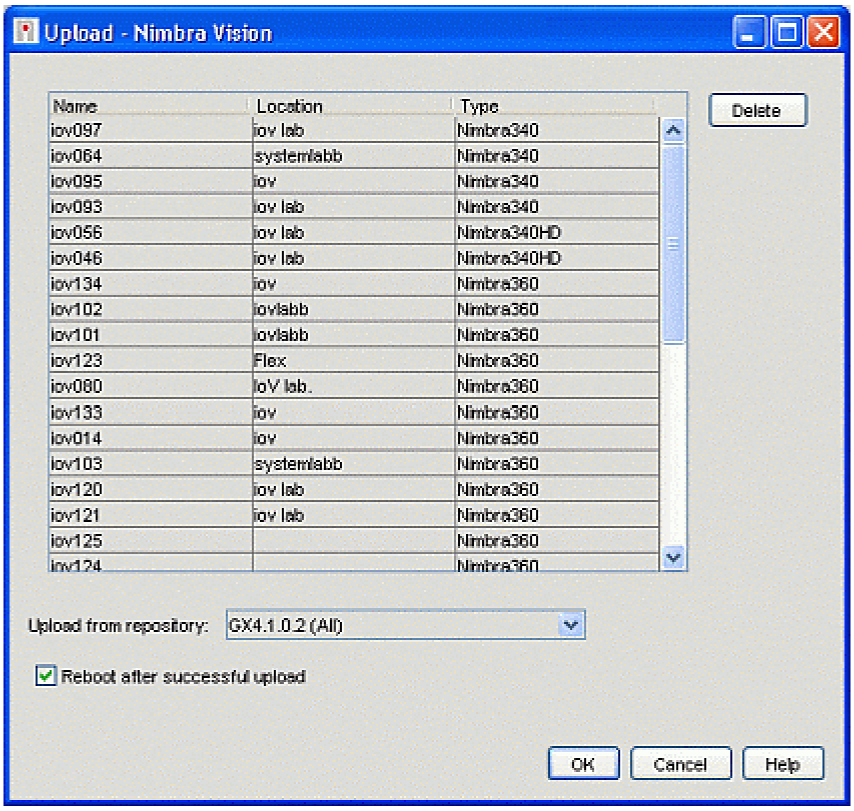
Name is a name of the repository. Enter any string that will describe the repository. The name will appear in the Upload Software/Firmware dialog.

URL is the Uniform Resource Locator for the repository. Specify the top folder for a single software and firmware distribution. Because it is the network element that is downloading the software and firmware from the repository, the URL entered must be on a format that can be used by the network element. The network element may not have DNS enabled, which means that the IP address may have to be used in the URL to the repository server. The protocols HTTP and FTP with anonymous login are supported. Some examples: `http://10.100.0.79:9090/repos/GX4.6.0.0`, `ftp://10.100.0.79/repos/GX4.6.0.0`.

Uploading Software/Firmware

To upload software and firmware to a node, select the network elements in the map or network database, and choose menu **Nimbra Node | Upload Software/Firmware…**. From here you can upload software and firmware, and you can restart the units (i.e. boards) that must be restarted to load the installed software or firmware.

© 2003-2010 Net Insight AB, All Rights Reserved



Name	Location	Type
iov097	iov lab	Nimbra340
iov064	systemlabb	Nimbra340
iov095	iov	Nimbra340
iov093	iov lab	Nimbra340
iov056	iov lab	Nimbra340HD
iov046	iov lab	Nimbra340HD
iov134	iov	Nimbra360
iov102	iovlabb	Nimbra360
iov101	iovlabb	Nimbra360
iov123	Flex	Nimbra360
iov080	IoV lab.	Nimbra360
iov133	iov	Nimbra360
iov014	iov	Nimbra360
iov103	systemlabb	Nimbra360
iov120	iov lab	Nimbra360
iov121	iov lab	Nimbra360
iov125		Nimbra360
iov124		Nimbra360

Upload from repository: GX4.1.0.2 (All)

[✓] Reboot after successful upload

Upload Software Firmware

The table with network elements lists the network elements that were selected when the dialog was opened. The table lists all the target network elements.

Upload from repository. This combo-box specifies from what repository to load software and firmware. If **No repository** is selected, then no software or firmware will be uploaded. Only images that are required are loaded from the repository. See Mange Software/Firmware Repositories on how to add repositories.

Reboot after successful upload. Select this checkbox to reboot units (boards etc.) that are running a different image then the image that is installed as primary image. When a unit starts, it will attempt to start with its primary image. Note that only the units that are not running the same image as the installed primary image are restarted.

To upload and restart the units with newly instilled images, select a repository and check **Reboot after successful upload**.

To upload, but restart at a later time you can run the function in two steps. As first step, select a repository but leave **Reboot after successful upload** unchecked. Primary and secondary images will then be installed, but the units are not restarted leaving the running image as it is. As second step, run the function again but check **Reboot after successful upload** to restart the units where the running image differs form the primary image. Because the new images were already installed, you can select **No repository**, but it is always safe to use the same repository as at the first step.

Click **OK** to begin the upload. This will open a progress window.

Click **Cancel** to exit the dialog without doing an upload.

Click **Delete** to delete any network element from the table.

The function will run for multiple network elements in parallel.

© 2003-2010 Net Insight AB, All Rights Reserved

Note! *Uploading takes several minutes. The progress dialog is updated with each progress step reported by the network element, but each step may still take several minutes.*

Note! *You must have a Node user defined for the network elements. See Node Users in the Administrator's Guide.*

See Also

Manage Software/Firmware Upload Repositories

Node Users in Administrator's Guide

Functional description of Software/Firmware Upload

The exact behavior of the upload process depends on the installation program located in the repository provided in the software and firmware distribution. Changing configuration parameters in Nimbra Vision can also alter the behavior.

The function starts the upload to multiple network elements in parallel. If the limit of the number of concurrent sessions is reached, then upload to the remaining network elements are queued, and will start when a session is completed.

If a repository is specified, then Nimbra Vision opens a telnet session to the network element, and logs in using the user name and password as specified in Node Users. On the network element, it issues a command to download the installation program the network element from the repository. The program is stored on a temporary folder on a volatile disk. The installation program is then started on the network element. The installation program examines what is already installed as primary and secondary software and firmware images. All images that do not match the contents (article number and version) in the repository are downloaded and installed as secondary image. When the installation is confirmed successful, it swaps the secondary and primary images, making the primary image the just installed image. It then downloads and installs the image again as secondary. This means that when complete, the primary and secondary images are both matching the contents of the repository.

If a repository has not been specified, i.e. **No repository** is selected, then this step is skipped.

If the **Reboot after successful upload** has been ticked, then Nimbra Vision opens a telnet session to the network element. It then starts the previously downloaded installation program with an option to restart units. The installation compares the running image with what is installed as primary image, and restarts all units (e.g. node controller, boards etc.) where the image differs. Note that reboot of the unit is only being done if the primary image is not the same as the running image.

The installation program is reporting progress for each step of the installation process to Nimbra Vision. The progress indicator in the Nimbra Vision client is updated as each progress step is reported. If an error would occur, this is also reported to the progress window.

In rare occasions you may want to have different primary and secondary images installed. One such occasion is when evaluating a potentially unstable distribution. If the unit crashes shortly after boot, it will load the secondary image, falling back to its behavior. To install only as primary images, prepend the URL with the word `--pri-only`. This will tell the installation program to only install the primary images.

© 2003-2010 Net Insight AB, All Rights Reserved

Warning! *It is only the faulting unit that loads the secondary image. The other units will still use its primary image. Internal compatibility between different products is not guaranteed across different distributions, meaning that a system running images from different (mixed) distributions may fail. Running images from mixed distributions is considered non-normal. Always contact Net Insight before installing mixed systems.*

© 2003-2010 Net Insight AB, All Rights Reserved

Administrator's Guide

Before You Start

Before You Start

Before you start Nimbra Vision the first time, you should do the following:

- Check the Release Notes and the latest Known Issues.
- **Setting up the Hosts Database.** Nimbra Vision uses host name lookup to match the nodes' IP addresses to host names.

You are now ready to start Nimbra Vision server. After you have started the server, you have to carry out the following:

- **Setting up the Nodes.** The nodes must send SNMP notifications to Nimbra Vision. It must also have a user (principal) that can be used by Nimbra Vision for its SNMPv3 commands, when configuring the node. The event history log size must also be correctly set. You can also do this before starting the server.
- **Setting up the Discovery Engine.** Nimbra Vision must be configured how to discover the nodes.
- **Setting up the SNMP Parameters.** If SNMPv3 discovery is not used, Nimbra Vision must be configured with the user credentials that it shall use when using SNMPv3 commands for configuring the nodes.

There are some other NMS configurations that should be considered. These can all be done after the server has started:

- Recommended NMS Configuration. Setting up of users, policies and maintenance.

Setting up the Nodes

Before Nimbra Vision can fully monitor elements, the nodes must have some of the SNMP parameters configured.

This chapter describes how to configure the Net Insight switches for use with Nimbra Vision. The operation is the same for all Net Insight switches, but may be different for different releases.

- Configure SNMP Notification Receivers. The node must send SNMP notifications to the management station. If redundant networks are used, notifications must be sent to the management station's both IP

© 2003-2010 Net Insight AB, All Rights Reserved

addresses. See Redundant Management Networks for details about how to use redundant networks.

- Configure the SNMP community and principal
- Increase the Event Log Size

Configure SNMP Notification Receiver

The node shall send SNMP notifications to Nimbra Vision. Notifications are sent when an alarm status or a configuration has changed.

Note! *If the node never sends any SNMP notifications, Nimbra Vision cannot detect that misses any SNMP notifications. The managed object will in this case not be updated with the latest status.*

Please consult the Nimbra documentation for details on how to configure the notification receiver.

Use the element manager or CLI to connect to the node, and configure the following parameters:

- The notification receiver. This is the IP address of Nimbra Vision server. On Net Insight's Nimbra nodes, you can configure multiple notification receivers. Notifications are sent to all the receivers. This means that you may have multiple NMS' monitoring a node.
- The SNMP port. This is the UDP port on Nimbra Vision that listens for SNMP notifications. The port shall be 162, which is standard for SNMP.

To set the notification receiver, the procedure is:

1. Open a web browser and connect to the node.
2. Login to the node.
3. Navigate to the SNMP configuration page using the left pane menu: **Control network | SNMP**.
4. Select the button **Add SNMP notification receiver…** to add a new notification receiver, or select an existing from the table if you want to modify an existing notification receiver.
5. Add/change the IP address and UDP port number.
6. Click **OK**.
7. Save the configuration when all the necessary configurations have been completed.

Configure the SNMP community and principal

Nimbra Vision uses SNMPv1 and/or SNMPv2c when monitoring the nodes. It may use SNMPv1/v2c or SNMPv3 when configuring the nodes. You have to configure the community name used for SNMP get operations.

The configuration procedure depends on the version of the switch's System Software. Please consult you switch documentation for details on how to configure a user login or principal.

To configure the community name and user name, the procedure is:

1. Open a web browser and connect to the node.
2. Login to the node.
3. Navigate to the SNMP configuration page using the left pane menu: **Control network | SNMP**.

© 2003-2010 Net Insight AB, All Rights Reserved

4. Set a read-only community name in the **Read-only community name** field. This community will allow SNMPv1/v2c read operations, but not write operations. Leave this blank if you want to use the same community name for read and write.
5. Set a read-write community name in the **Read-write community name** filed. This community will allow both read and write operations using SNMPv1/v2c. You can leave this blank to disable write operations.
6. Set an SNMPv3 user name in the **SNMPv3 user** field. This is only necessary if SNMPv3 shall be used.
7. If SNMPv3 authNoPriv shall be used as security level, set the **Authentication key** as the password to use for authentication.
8. If SNMPv3 authPriv shall be used as security level, set the **Privacy key** as the password to use for data encryption.
9. Click **OK**.
10. Save the configuration when all the necessary configurations have been completed.

Increase the Event Log Size

The event log in the node should be large enough to hold event for at least a couple of minutes. Nimbra Vision uses this event log to fetch events that has been lost. If the log does not contain the missed event, Nimbra Vision will have to refresh the complete node. This is an operation that is more expensive.

A good rule of thumb is that the event log size should be at about 2.5 times the number of expected connections, but not less than 50.

You must also ensure that all applicable events are sent to Nimbra Vision.

The procedure to set the event log size is:

1. Open a web browser and connect to the node.
2. Login to the node.
3. Navigate to the system page using the left pane menu: **Maintenance | System**.
4. Change the value of Event log size.
5. Ensure that the at least the following events are logged: *performance*, *info*, *config* and *alarm*.
6. Click **OK**.
7. Save the configuration when all the necessary configurations have been completed.

See Also

Redundant Management Networks

Setting up the Host Database

When Nimbra Vision discovers a new node, or makes a refresh (rediscovery) of the node, it gives it a name based on the IP address of the node -- it gives the managed object the host name. It will use the node's IP address to lookup the name. This function depends on your server's operating system.

For correct function of Nimbra Vision, the name lookup must report the same result each time for every IP address of the node. If no name is returned by the function, the IP address is used as a name. If another name is returned, a new

© 2003-2010 Net Insight AB, All Rights Reserved

object will be created in the database, populating the database with multiple managed objects that represent the same node.

Normal operation of the host name lookup is:

1. Query the local host database, if not match is found on the IP address, then
2. Query the DNS server, if one is used.

For Windows NT4/2000, the local host file is named `C:\WINNT\system32\drivers\etc\hosts`.

For Windows XP, the file local host file is named `C:\WINDOWS\system32\drivers\etc\hosts`.

If redundant management networks are used, the hosts file must be used, you can not use DNS.

This hosts file contains the mappings of IP addresses to host names. Each entry should be kept on a separate line. The IP address should be placed in the first column followed by the corresponding host name. The IP address and the host name should be separated by at least one space. Additionally, comments may be inserted on individual lines or following the host name denoted by a '#' symbol.

If a node has multiple IP addresses, there shall be one entry per IP address. The order of the entries shall be according to the priority in falling priority order. I.e. Nimbra Vision will use the first IP address for a node specified in the file, that is reachable.

Example:

```
# This is an example host file
127.0.0.1      localhost

192.168.1.1    node01       # primary address
192.168.200.1  node01       # secondary address

192.168.1.2    node02
```

You can modify the hosts file while Nimbra Vision is running. Nimbra Vision will periodically check if the file has been modified (default is 2 minute intervals).

Setting up the SNMP Parameters

SNMPv1/v2c or SNMPv3

Nimbra Vision can use SNMPv1/v2c or SNMPv3 for its communication with the nodes. SNMPv3 supports authentication and encryption, but use of this is slower, and user name and password must be configured for each node.

The default configuration of Nimbra Vision is to use SNMPv2c for read operations used when monitoring the node (SNMP-GET), while making it possible to use SNMPv3 with authNoPriv (e.g. authentication but no encryption) for all write operations (SNMP-SET). It is recommended to use SNMPv2c for read and SNMPv3 with authNoPriv for write operations.

A community name is a string, much like a password, that must be agreed upon between Nimbra Vision and the node. Note that the community name is stored in clear text, and is also transported in the SNMP protocol as clear text. It should therefore not be used as a means of authentication. For read operations using SNMPv1/v2c, the read community name is used. If SNMPv2c is to be used for write, then the write community name is used.

© 2003-2010 Net Insight AB, All Rights Reserved

If SNMPv3 is used with authentication (authNoPriv) or encryption (authPriv), then a user name (principal) and password (key) is used. The password is encrypted.

In Nimbra Vision, the read and write community names, the user name and the SNMP version to use are stored in the managed object representing the node. The remaining settings used for SNMPv3 are stored is a separate area in the database, the SNMP V3 Security database. You can use the **SNMP V3 Security** configurator located under the **SNMP Tools** node in the Tree to access the data. For each combination of node and user, an entry shall exist in the table. Use the button **Refresh** to update the displayed table with data from the database. Nimbra Vision stores the SNMPv3 parameters for each node and user name combination.

The SNMP settings in the node and in the database are set or updated by the Discovery Engine at node discovery or at refresh. You should therefore setup the Discovery Engine so that it has correct settings for SNMPv1/v2c or SNMPv3 for all IP interfaces on all nodes managed by Nimbra Vision. See Setting up the Discovery Engine.

When Nimbra Vision makes write operations, it will use the SNMP version as configured in the managed object. If SNMPv3 is used, then the user name as set in the managed object is used to lookup the security setting in the SNMP V3 Security database. If SNMPv1 or SNMPv2c is used, then the write community as set in the managed object is used.

When Nimbra Vision makes read operations, it can use a different SNMP version then what is set in the managed object. The parameter SNMP_GET_VERSION in the file `<NMS_Home>\conf\serverparameters.conf` specifies what version to use; 1, 2, 3 or 0 to use the version as set in the managed object. The default value is 2, i.e. to use SNMPv2 for read operations.

SNMP Timeout and Retries

Because SNMP is using UDP, packet may be dropped or otherwise lost on the network. Nimbra Vision has functionality to recover when SNMP packets are lost. If it fails to get a response on an SNMP query, it will try again a couple of times before giving up and assuming that the node is unreachable.

You can configure the timeout for the reply before it is assumed that an SNMP packet is lost, and the number of retries before giving up. The timeout is specified in seconds and applies for the first timeout. If it still fails to get a response, then the timeout is doubled for each retry.

The Discovery Engine is responsible for discovery of new nodes. Use the Discovery Configurator to set the parameters used during discovery. These values will considerably affect the time it takes to discovery a new network. If the time is set to short, it will fail to correctly discover the node, but if it is set to long (many retries and/or long timeout) then it will take long time before giving up an IP address that is not used.

- SNMP Retries
- SNMP Timeout

The Status Poller (tester) periodically tests the connectivity to the node. Configure the parameters in the file `<NMS_Home>\conf\serverparameters.conf`. The file is read when the server is started.

- STATUS_POLL_SNMP_RETRIES
- STATUS_POLL_PING_TIMEOUT (there is no dedicated parameter for SNMP, the parameter of Ping is instead used).

© 2003-2010 Net Insight AB, All Rights Reserved

For Data Collection of performance data, the settings are made in the file `<NMS_Home>\conf\NmsProcessesBE.conf`. The file is read when the server is started. Configure the parameters for the process `com.adventnet.nms.poll.Collector`.

- SNMP_COLLECTION_SNMP_RETRIES
- SNMP_COLLECTION_SNMP_TIMEOUT

See Also

Setting up the Discovery Engine

Setting up the Discovery Engine

You must configure how Nimbra Vision shall discover the nodes in the networks. The Discovery Engine has a set of rules and seeds that are used by Nimbra Vision to discover the nodes that you want Nimbra Vision to monitor.

The configuration is easiest done using the Discovery Configurator, which is described in the WebNMS Administrator's Guide. The Discovery Configurator can be started from Nimbra Vision client: select menu **Tools | Runtime Administration** and select **Topology | Discovery Configurator**.

This topic describes some tips on how to set-up the discovery engine. First read the topic Setting up the SNMP Parameters for details about the SNMP parameters.

The headings below refer to the different tabs in the **Discovery Configurator** application:

- The General Tab - general settings for the discovery engine
- The Protocol Tab - configuration of general but protocol specific parameters
- The Network Discovery Tab - configuration of IP networks or address ranges to discover
- The Node Discovery Tab - add individual nodes as seeds prior discovering networks or ranges
- The Criteria Tab - limit discovery to only include nodes having specified characteristics, e.g. only Nimbra nodes

Note! *To setup the discovery engine for using SNMPv3 discovery, Nimbra Vision server must be running (actually, only the Nimbra Vision database must be running).*

The "General" Tab

This tab contains general settings for the discovery.

- **AutoDiscover**. This shall be enabled. This controls if Nimbra Vision shall discover nodes or not.
- **Rediscover Already Discovered**. This shall be disabled. The discovery engine makes periodic scans of the networks to search for new nodes. This controls whether nodes already known by Nimbra Vision shall be discovered again as if they were not discovered before.
- **Discover LocalNet**. This shall be disabled. This controls whether the IP network where the Nimbra Vision server is located shall be discovered

© 2003-2010 Net Insight AB, All Rights Reserved

or not, without requiring to specify the IP network. It is not possible to make a full SNMPv3 discovery using this. Specification of all discovered IP networks should be made under the Networks Discovery tab or the Node Discovery tab.

- **Discovery Interval**. Recommended value 1. The Discovery Interval in seconds between discover of two devices. Default is 1 second. This is **not** used during the first discovery after the server has started, only during scheduled rediscoveries. To set the interval for the first discovery, open the **Initial Parameters** form.
- Configure the rediscovery scheduler from the **Rediscovery**. The rediscover will discover new nodes. If the network is static, the discovery interval could preferably be set to long. The default setting of 24 hours should be adequate, but if your network is static, you could increase this value.

Note! *You can manually force a rediscovery of a network. This could be done if e.g. you have added nodes to the network and do not want to wait for the scheduled rediscovery. To force rediscovery, select the network in the IP map or in the Network Database and select menu* ***Network | Stop Discovery*** *followed by* ***Network | Start Discovery****.*

You can also manually add a single node or an IP network: from the Network Database select the menu ***Edit | Add Node*** *or* ***Edit | Add Network****. When the node is added, the parameters for the Discover engine will be used.*

The "Protocol" Tab

This tab allows for configuration of all management protocols supported by Nimbra Vision. For additional information about SNMP parameters, see Setting up the SNMP Parameters. The discovery engine will try in order SNMPv3, SNMPv2, SNMPv1 and finally ICMP ping.

For the **SNMP** protocol, the following shall be configured:

- **SNMP Discovery**. Shall be enabled. All Nimbra nodes shall be discovered using SNMP.
- **SNMP Retries**. Recommended value is 1. The number of retries if the discovery engine does not get a reply to SNMP queries. Because SNMP packets may be lost, the discovery retries if it fails to get a reply. If no reply is received, then the discovery engine assumes that no SNMP agent is available on the IP address.
- **SNMP Timeout**. Recommended value is 3. The number of seconds to wait for the first reply before assuming it is lost. The delay is doubled for each retry.
- **Read community**. The default value used as community name for SNMPv1/v2c read operations. If SNMPv1/v2c discovery is done, then this value will be used as read community. See Network Discovery tab and Node Discovery tab below for how to override the default value. When SNMPv3 discovery is done, then this value will be set as read community on the managed object for the discovered node. The value should be set to match the read community in the nodes.
- **Write community**. The default value used as write community name for SNMPv1/v2c write operations. If SNMpv1/v2c discovery is done, then this value will be set as write community unless otherwise specified in Network Discovery tab or Node Discovery tab below. When SNMPv3 discovery is done, then this value will be set as write community on the managed object for the discovered node. The value should be set to match the value in the nodes.
- **SNMP Ports** shall be 161, which is the standard port for SNMP.

© 2003-2010 Net Insight AB, All Rights Reserved

- **SNMPv3 Discovery** must be enabled when SNMPv3 discovery shall be done.
- **User Names**. Should be empty. This is the default value used as user name in case of SNMPv3 discovery when discovery is done with security level *noAuthNoPriv*. It is not used when doing discovery with security levels *authNoPriv* or *authPriv*.
- **Context Name**. Shall be empty. This is the default value used for context name.

For the **ICMP** protocol, the following shall be configured:

- **ICMP Discovery**. Recommended but not necessary to be enabled. This controls if discovery shall be done using ICMP ping. If SNMP discovery fails (or is disabled), then the Discovery Engine makes an ICMP discovery if this is enabled. SNMP discovery will fail if the SNMP settings in the Discovery Configurator do not match the settings in the node. When ICMP discovery is enabled, the node will be discovered anyway and the configuration error can be found. Nimbra Vision will not be able to determine the type of a node that is discovered using ICMP, and the node will therefore be added as a managed object of type *Node*.
- **Ping Retries**. Recommended value is 1.
- **Ping Timeout**. Recommended value is 0.

The "Network Discovery" Tab

This tab allows for configuration of discovery for IP networks or ranges of IP addresses. You can use this in combination with Node Discovery (see below). If only a very few nodes exist on a network, then it might be easier to configure only using the Node Discovery tab. For each network or range of IP addresses, configure the following:

- **Discover**. Normally enabled. You can prevent a network or IP addresses within a network from being discovered by disabling this. If disabled, then no managed object will be created for any addresses in this network.
- **Entire Network** or **Set of Nodes**. Select as appropriate. If only a part of the network contains nodes that shall be discovered, then the discovery will be completed faster if a range is specified (Set of Nodes). This is because if a node does not exist on an IP address, then the Discovery Engine will try SNMP (with retires) followed by ICMP Ping (with retries) before concluding that the IP address is not used.
- **IP Address** is the network address.
- **NetMask** is the network mask.
- **Start IP** and **End IP** are the first and last IP address if only a range of addresses is to be discovered.
- **DHCP** is not supported for Nimbra nodes. Uncheck the **DHCP** checkbox.

You also need to configure SNMP parameters. The network could be discovered using SNMPv1, SNMPv2c or SNMPv3. See Setting up the SNMP Parameters for details. If SNMPv3 is used, or if SNMPv1/v2c where other settings than the defaults configured at the Protocol tab (see above) are to be used, then you need to enable the **SNMP** checkbox and **SNMP Properties**. Here, add a list of configurations that will be tried in order. If SNMPv3 discovery is to be done, add configuration as described in the following table:

© 2003-2010 Net Insight AB, All Rights Reserved

Parameter	Description
UserName	The user name (principal) as configured in the node(s).
Context Name	Shall be empty. Nimbra nodes use the default context, which is the empty string.
Port	This is always 161. The UDP port to use for SNMP.
Security Level	*AuthNoPriv* or *AuthPriv* as configured in the node.
Auth Protocol	*MD5*, as configured in the node.
Auth Password	The password (authentication key) for SNMPv3 authentication, as configured in the node(s)
Priv Protocol	*CBC-DES*. This shall only be set when security level is *AuthPriv*.
Priv Password	The password (privacy key) for SNMPv3 encryption, as configured in the node(s). This shall only be set if the security level is *AuthPriv*.

***Note!** If a node has multiple IP addresses, as would be the case if a redundant management networks are uses (see Redundant Management Networks), then the Discover engine should be configured for all these IP addresses.*

The "Node Discovery" Tab

Use this tab to add individual nodes for the discovery process. Entries entered here will be discovered before the networks configured at the Network Discovery tab (see above) are being discovered. This means that you can also use this to override the network settings for individual nodes. Added nodes will be used as seed, and their knowledge of the network will be used for further discover.

- **Discover**. Normally enabled. This checkbox controls if the IP network shall be discovered, or if discovery of IP address is suppressed. If disabled, then no managed object will be created for this IP address.
- **IP Address(es)**. The IP address of the node to be discovered.
- **NetMask**. The netmask of the node to be discovered.
- **SNMP Version**. Select *v3* for SNMPv3 discovery. See Setting up the SNMP Parameters for details.
- **Discover Parent Net**. Shall be disabled to prevent the complete IP sub-network containing the node to be discovered. If this is enabled, discovery of the network will be done using the settings configured in the Protocol tab (see above).
- **Community**. Not used for SNMPv3 discovery. This is the read community name in case of SNMPv1 or SNMPv2c discovery.
- **SNMP AgentPort**. Shall be 161. The UDP port to use for SNMP.
- **UserName**. The user name (principal) to use at SNMPv3 discovery.
- **ContextName**. Shall be empty. Nimbra network elements use the default contexts, which is the empty string.
- **Properties**. When doing SNMPv3 discovery, you also need to configure the parameters displayed when clicking on the **Properties** button. The parameters are the Security Level, Auth Protocol, Auth Password, Priv Protocol and Priv Password. These shall match as configured in the node.

© 2003-2010 Net Insight AB, All Rights Reserved

The "Criteria" Tab

Use this tab to setup constraints for discovering individual nodes. E.g. if your network(s) contain other types of nodes that you do not want to discover, you can define these constraints here.

To only discover Nimbra nodes, set the constraint as:

- **Allow Criteria**. Enable this. Only nodes matching all the criteria will then be discovered (and added to the database).
- **Property Name**: *type*
- **Property Value**: *Nimbra**

If such a criteria is used, then it makes no sense to enable ICMP discovery at the Protocol tab (see above).

See Also

Setting up the SNMP Parameters

Discovering Your Network in the Web NMS Administrator Guide

Management Networks

Redundant Management Networks

Using redundant management networks ensures that no single point of failure in the management network will result in that Nimbra Vision loses contact with the nodes.

When using redundant management networks, the nodes have multiple IP addresses and are connected to different management networks. The different networks can be implemented as different in-band management networks, or as a combination of in-band and out-band management networks.

See Also

Designing Redundant Networks

Setting up Nimbra Vision for Redundant Networks

Designing Redundant Networks

The redundant management networks can be implemented either using multiple in-band management networks, or as a combination of in-band and out-band management networks. Generally, the idea is to design the networks so that each node is connected to two management networks. The networks are designed in such a way that a failure in one of the management networks will not affect the other. The node will have two IP addresses, one at each network. The Nimbra Vision server, being the management station, will also have two IP addresses, and is possibly using two network interface cards.

The IP routing in the node shall be setup so that IP traffic to the primary management station address uses the primary network, and IP traffic to the secondary management station address uses the secondary network. Similarly, the IP routing in the management station must be configured so that when a node's primary IP address is used, the IP traffic shall be routed through the

© 2003-2010 Net Insight AB, All Rights Reserved

primary network, and when a node's secondary IP address is used, the IP traffic shall be routed through the secondary network.

The node will have to send SNMP notifications (e.g. alarm notifications) to Nimbra Vision's both IP addresses. This will ensure that Nimbra Vision will receive the notification even if one of the management networks would be down. Nimbra Vision will silently drop received traps that are duplicates.

Setting up Nimbra Vision for Redundant Networks

When setting up Nimbra Vision for redundant networks, the following must be done:

- The management station must be assigned two IP addresses. The management station should preferably be equipped with two Network Interface Cards (NIC) to ensure complete network redundancy.
- The IP routing must be configured with entries on how to reach the primary and secondary IP addresses of the nodes. The routes should use the different networks, and hence the different NIC's.
- The hosts database must have entries for each IP address and node. The hosts file shall have one entry for each IP address. This means that more than one IP address will be assigned to one host name; one entry for each redundant IP address. The entries in the hosts file shall be ordered in priority order. Nimbra Vision will use the first entry in the hosts file that matches the node, and that has connectivity. See Setting up the Host Database.

See Also

Setting up the Host Database

Redundant Network Example

How does Nimbra Vision use Redundant Networks?

Redundant Network Example

How does Nimbra Vision use Redundant Networks?

Each node that is managed by Nimbra Vision is represented by a managed object (MO) in Nimbra Vision database. The name of the managed object is the same as the hostname for the managed object, as defined in the hosts file. A name of the managed object can be mapped to multiple entries in the hosts file, one for each IP address that is used by the node.

When Nimbra Vision shall access a node, it makes a name lookup to see what IP address to use for the managed object with a specific name. The managed object property *activeIpAddress* is always updated to reflect the IP address that is currently used to reach the node.

When a managed object representing a node is added to the database, Nimbra Vision will lookup the IP address and set the managed object property *name* to the corresponding host name. The property *ipAddress* will be set to the IP address that was used to discover or add the managed object. A refresh operation on a node is like a rediscovery of the node, meaning that the property *ipAddress* will be set to the IP address currently used to access the node.

© 2003-2010 Net Insight AB, All Rights Reserved

Nimbra Vision is periodically checking the status of the node. In this process, Nimbra Vision tests for DtmNode managed object which IP address has the highest priority and is reachable, and will update the DtmNode managed object property *activeIpAddress* accordingly. The tester will generate an alarm and indicate that the node is unreachable when all IP addresses fail, and will clear the alarm when any IP address is reachable. In addition, Nimbra Vision will detect that an IP address is unreachable for some additional SNMP operations.

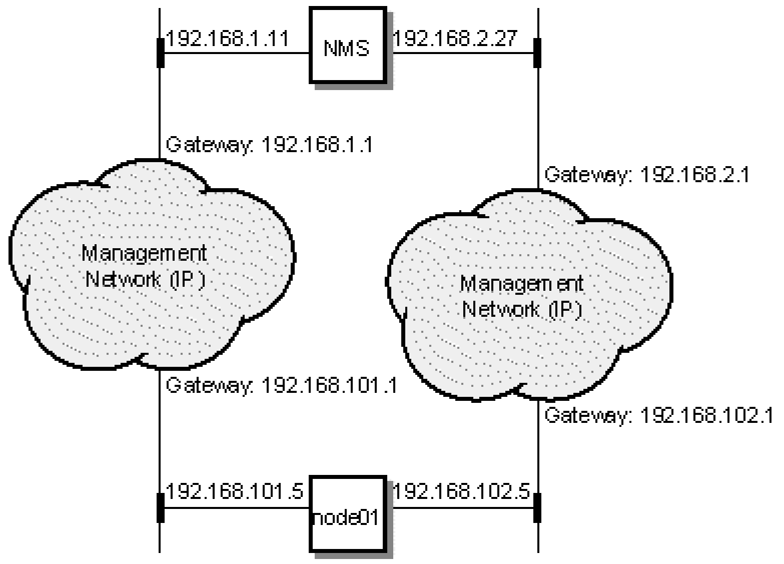
Nimbra Vision is also periodically checking the status of all the IP interfaces, and will generate an alarm to indicate that the IP address is unreachable when it fails to reach the interface. The tester will clear the alarm next time the interface is tested and is successfully reached.

The hosts file is periodically checked to see whether it has been updated. If the file has been updated, the information in Nimbra Vision is updated from the file. The new IP address to hostname and priority order is used.

For full redundancy, the DtmNode nodes should send each event as a SNMP notification (SNMPv2 traps) to each of the management stations IP addresses. This means that the notifications will take different routes through the management networks, ensuring that at least one of the notifications will reach the management station even if one of the management networks are failing. When Nimbra Vision receives a notification (SNMPv2 trap) from a DtmNode node, its sequence number is examined. If the notification has already been received (i.e. from another IP address representing the same node), the notification is discarded.

Redundant Network Example

This example shows a configuration of a management station and a node interconnected using two redundant networks.



Example network with one management station (NMS) and one node (node01) interconnected using redundant IP networks.

The management station is equipped with two Network Interfaces, connected to the two IP subnets 192.168.1.0/24 and 192.168.2.0/24. Each IP interface is assigned one IP address each: 192.168.1.27 and 192.168.2.11.

© 2003-2010 Net Insight AB, All Rights Reserved

The gateway for the network 192.168.1.0/24 is 192.168.1.1, and for 192.168.2.0/24 is 192.168.2.1.

A DtmNode node, named *node01*, is connected to two networks, one in-band and one out-band management network. The in-band management network address is 192.168.101.0/24, and its gateway is 192.168.101.1. The out-band management network address is 192.168.103.0/24, and its gateway is 192.168.102.1. The in-band IP address is set to 192.168.101.5. The physical interface is connected to the out-band network, and the IP address is set to 192.168.102.5.

The routing must be setup between the networks 192.168.1.0/24 and 192.168.101.0/24. Similarly, routing is setup between the networks 192.168.2.0/24 and 192.168.102.0/24. IP traffic can now be sent to/from the networks 192.168.1.0/24 and 192.168.101.0/24, and to/from the networks 192.168.2.0/24 and 192.168.102.0/24.

The management station routing is configured so that IP packets can be sent to the node using its two IP addresses:

Destination: 192.168.101.0/24, Gateway: 192.168.1.1
Destination: 192.168.102.0/24, Gateway: 192.168.2.1

Two entries are added in the hosts file for *node01*:

```
# This is the hosts file
192.168.101.5   node01   # Primary address
192.168.102.5   node01   # Secondary address
```

The node routing is configured so that IP packets can be sent to the management station using its two IP addresses:

Destination: 192.168.1.0/24, Gateway: 192.168.101.1
Destination: 192.168.2.0/24, Gateway: 192.168.102.1

Two SNMP notification receivers are configured in the node to enable that notifications are sent using the two networks in parallel:

Notification receivers: 192.168.1.27, 192.168.2.11.

Security Management

Security Management

Security Administration helps you to manage Nimbra Vision security information. Administrators can achieve Fine-Grained Authorization by setting scope for the operations assigned to the group. This scope will define the restricted access for the operation in that group. Also by setting Custom View Scope to groups, the users can be restricted to view only the required information in the client over which they can do the allocated operations. Thus setting the custom view scope criteria for a group of users to a particular network type, allows the users of the respective group to view only the nodes of that particular network on which the user can perform the authorized operations.

The Security Management in the Web NMS Administrator Guide describes how to configure the users, groups and their permissions.

Users

A user represents the person or entity that logs in to Nimbra Vision. A user must supply a password when logging in. A user can be assigned groups and permissions that describe what a user is allowed to do.

© 2003-2010 Net Insight AB, All Rights Reserved

User Groups

A group is a logical collection of users grouped together to access common information or perform similar tasks. A user can be a member of multiple groups that represents different roles. Each group can have different permissions.

Nimbra Vision comes pre-defined with a set of groups. The groups have assigned permissions for their roles.

- **SystemAdmin** Administration of the Nimbra Vision system. Manage and configure Nimbra Vision. Add and delete nodes (and managed objects).
- **NetworkAdmin**. Administration of the network. Manage and configure discovered nodes and managed objects.
- **ServiceAdmin**. Administration of services, including add/delete of services (service provisioning).
- **ServiceUser**. Retrieve and view detailed information about services.
- **Operator**. View status and configuration of the nodes and the network, pick-up alarms.
- **User**. Only view information that exists in Nimbra Vision.

Coarse-Grained Authorization

An operation represents something (an activity) that the user can be granted or denied. You can assign permissions to a group or user to grant or deny a group or a user to perform an operation. Operations are organized in a tree structure with parent and child operations. In this tree structure, operations can be included or excluded for a user or group.

If an operation is *included*, then the group or user is granted the permission to perform the operation.

If an operation is *excluded*, then the group or user is denied the permission to perform the operation.

If an operation is *not set*, then the operation will inherit the permission of its parent operation, and the group or user is granted or denied permission to perform the operation as specified on the parent operation (or its parent if not set).

The granting of an operation always precedes the denial of an operation. A user can be assigned multiple groups where an operation may be included in one group and excluded in another, and permissions can also be assigned on the user. In case of conflicting permissions, the operation would be granted.

Fine-Grained Authorization

Authorized Scope

With authorized scopes, it is possible to specify whether an operation is granted or denied for specific objects. As described about the coarse-grained authorization, an operation can be granted or denied for a group by *including* or *excluding* an operation. The group is then granted or denied permission for the operation for all objects. But, when an authorized scope is assigned to the operation for a group, then the group is granted the permission only if the object is matching the criteria specified for the authorized scope. It is thus possible to grant or deny the operation depending on the properties of the object. This is called fine-grained authorization. An authorized scope can only be assigned to operations that are *included* in a group; it cannot be assigned to *excluded* operations or to operations assigned directly to a user.

The authorized scope contains a set of properties with match strings. When an object is checked against the authorized scope, then the properties in the objects

© 2003-2010 Net Insight AB, All Rights Reserved

is matched against the properties specified in the authorized scope. All properties specified in the authorized scope must match. When defining match criteria for an authorized scope, then the exact matching string must be specified; wildcards etc. are not permitted. You can list alternative in the criteria, where the strings are separated with comma (`,`). The object's property must match any of the strings listed in the property to be granted.

The operation is granted for the object if any of the authorized scopes in any of the assigned groups for the user would grant the operation. Note that if an operation would be set as *included* in coarse-grained authorization (i.e. when no authorized scope is assigned the operation), then all objects would match.

See descriptions of the operations for more information about objects that are related to operations.

Custom View Scope

The custom view scopes are filters that control what objects a user is granted to view in the client. Only objects matching a set of criteria can be displayed in the client. Custom view scopes are defined per module; you can control what objects that can be displayed in the different modules:

- Network Database: Managed objects in the Network Database.
- Maps: Managed objects in maps; would reflect symbols in maps.
- Alerts.
- Events.
- Stats Admin.

Custom view scopes are assigned to groups. When custom view scopes have been assigned to a group, users assigned the group is enabled to view the objects matching the custom view scope criteria. The object must match all the criteria in a custom view scope. When a group has been assigned multiple custom view scopes in a module, then objects are displayed when they match any of the custom view scopes. If a user is a member of multiple groups, then the objects are displayed when they match the custom view scopes of any of the groups.

If none of the users' assigned groups of have custom view scopes assigned, then the user is granted to view all the objects.

When defining match criteria for a custom view scope, the same type of operations can be used as for Custom Views, such as wildcards, not-operator, or-operator, and-operator, and range operator. See Writing Filter Criteria.

See Managing Custom View Scopes in the Web NMS Administrator Guide for a guide on how to work with custom view scopes.

See Also

Security Management the Web NMS Administrator Guide

User Operations

User Operations

An operation represents something (an activity) that the user can be granted or denied. Operation can also be referred to as permission. Operations are organized in a tree structure with parent and child operations. Operations are assigned to groups or users, and control the permissions of the user.

© 2003-2010 Net Insight AB, All Rights Reserved

The Operation Tree contains a list of operations that are provided by default in Web NMS. Assigning different operations to different users is an administrative function.

Operations are divided into the following groups (branches in the tree), which all contains fine grained operations:

- Administrative Operations.
- Events
- Topology
- Policy
- User Administration
- Trap Parsers and Filters
- Alerts
- Configuration
- Maps
- Polling Units
- Polling Objects
- Threshold Objects
- Poll Filters
- DTM Network Access
- Service Provisioning
- Inventory
- Application Tree

In addition to the default operations defined in Web NMS, the following operations are defined:

Topology

Operation	Description	Fine-Grained Properties
Set Tag	Set tag property on managed objects	Properties of the applicable ManagedObject (e.g. DtmNode, DtmInterface, DtmLink, DtmService, TvgvNode, etc.).

DTM Network Devices

Operation	Description	Fine-Grained Properties
Telnet Remote Node Access	Open a telnet connection to the remote node.	Fine-grained authorization not supported for the operation.
Web Remote Node Access	Open a web browser and connects to a node.	Fine-grained authorization not supported for the operation.

© 2003-2010 Net Insight AB, All Rights Reserved

Operation	Description	Fine-Grained Properties
Ping Remote Node	Send ICMP Ping to the remote node.	Fine-grained authorization not supported for the operation.
Trace Route Remote Node	Displays the route in the IP network to the remote node.	Fine-grained authorization not supported for the operation.
Get System Information	Read and display the basic system information from a node.	Properties in DtmNode managed object.
Modify System Information	Change the setting of the basic system information in a node, such as sysName, sysContact and sysLocation.	Properties in DtmNode managed object.
Get DTM Interface	Read and display current settings of the DTM interfaces in a node.	Properties in DtmInterface managed object when accessing a single interface, or properties in DtmNode managed object when listing interfaces on a node.
Modify DTM Interface	Change the settings of a DTM interface in a node.	Properties in DtmInterface managed object.
Get Node Configurations	Read and display the status of the persistent saved configurations on a node.	Properties in DtmNode managed object.
Modify Node Configuration	Change the settings or configuration of the persistently saved the configuration locally in the node and backed-up in Nimbra Vision.	Properties in DtmNode managed object.
Upload Software	Work with and update Software and Firmware on the nodes.	Properties in DtmNode managed object.
Trace Sync	Trace the network synchronization.	n/a
Trace Channel	Trace a channel in the network	If the service is known, then properties in DtmService managed object, otherwise properties in DtmNode managed object of the originating node.
List Channels	List Channels through a node.	Properties in DtmNode managed object.
Edit Custom View	Editing settings of custom views.	n/a

© 2003-2010 Net Insight AB, All Rights Reserved

Service Provisioning

Operation	Description	Fine-Grained Properties
Create Service	Create a new service or create a destination to a service.	- **name** - **ttpCustomerId** - **ttpMode** - **ttpServiceType** - **type** - **origDtmNode** - **origAccessInterface** - **tag** (as originally inherited from originating node) - **termDtmNode** - **destinationTag** (as inherited from terminating node) **Note:** The use of the value `NULL` for a value on the terminating side will match when the service has no destination, see below.
Get Service	Get and display information about a service.	Properties in DtmService managed object. And the following property: - **destinationTag** (this is the current value of **tag** at the destination DtmNode managed object) Note: The use of the value `NULL` for a value on the terminating side will match when the service has no destination, see below.
Modify Service	Change settings in a service, including change of associated destination.	
Delete Service	Delete a service or delete a destination on a service.	
Preemption	Access the preemption function	n/a
Redundant Headend	Add or edit redundant headend object.	Properties in DtmService for both services in Redundant Headend object for update.
Create Forwarding Function	Create a forwarding function.	Properties in DtmNode managed object. And the following properties: - **ffName** (for ETSv1, simulated as the first and only forwarding function on the board). - **customerId** (ETSv2 only) - **purpose** (ETSv2 only)
Get Forwarding Function	Get and display information about a forwarding function.	
Modify Forwarding Function	Change settings in a forwarding function.	
Delete Forwarding Function	Delete a forwarding function.	
Get Access Interface	Read and display settings of access interfaces	Properties in DtmNode managed object.

© 2003-2010 Net Insight AB, All Rights Reserved

Operation	Description	Fine-Grained Properties
Modify Access Interface	Change setting of the access interface.	And the following properties: - **ifName** - **customerId** (ETSv2 only) - **purpose** (ETSv2 only) - **forwardingFunction** (ETSv2 only)
Get Source Route	Read and display information about a source route.	Properties in DtmNode managed object where the source route is defined. And the following property: - **routeName**
Modify Source Route	Change the settings of a source route.	
Use Source Route	Use the source route in a service.	

Services without Destinations

A service may or may not have a destination. For Create/Get/Modify/Delete Service operations, it is therefore necessary to also be able to specify fine-grained authorization when there is no destination. For the Create/Get/Modify/Delete Service operations, the following properties of the service object will have the value `NULL` when the service has no destination: **termDtmNode**, **termAccessInterface**, and **destinationTag**. Use `NULL` to match this.

Example:

The Create Service operation has a scope where the property **destinationTag** has the value `team-A`. The users will be able to create services that terminate on nodes that are tagged with team-A. But the user cannot create a service that does not have a destination because the **destinationTag** of the service object will not match `team-A`. But, if the scope is changed so that **destinationTag** has value `team-A,NULL`, then the user will also be able to create a service with no destination because the **destinationTag** of the service object will have value `NULL`.

Inventory

Operation	Description	Fine-Grained Properties
Search Inventory	Search the inventory database	The node of the inventory object must match Custom View Scope in the Network Database module.
Browse Inventory	Browse the inventory database	

Applications Tree

You can enable and disable nodes in the Applications Tree in the left pane of the user interface. The following operations correspond to different nodes in the tree.

Operation	Description
Network Map Tree Node	This operation controls access to the node containing maps both IP and DTM maps.
Fault Management Tree Node	This operation controls access to the node with alarms and events.

© 2003-2010 Net Insight AB, All Rights Reserved

Operation	Description
Configuration Tree Node	This operation controls access to the node with the audit trail.
Performance Tree Node	This operation controls access to the node with poller objects and collected statistics.
Network Database Tree Node	This operation controls access to the node with the Network Database.
Administration Tools Tree Node	This operation controls access to the node with policies.
SNMP Tools Tree Node	This operation controls access to the node with SNMP tools, such as SNMP browser.

See Also

Operations Tree in the Web NMS Administrator Manual

Running Behind a Firewall

When running Nimbra Vision server or client behind a firewall, it is necessary to open ports in the firewall to enable communication between the server and the nodes, and between the server and the client. The firewall must support stateful inspection to be able to identify the sessions initiated from behind the firewall.

This topic describes how to do when

- Running Nimbra Vision server behind a firewall
- Running the nodes behind a firewall

Running Nimbra Vision Server Behind a Firewall

This section explains how Nimbra Vision can be started behind a firewall. Firewalls act as barriers preventing unauthorized access to a network. Firewalls are used for security purposes. Nimbra Vision can run behind a firewall. This facility will enable to authorize access to the network where Nimbra Vision server is located to some, but not to others.

To run Nimbra Vision behind the firewall, the ports used to access Nimbra Vision must be opened. The table below lists the ports that are used to access Nimbra Vision server.

Port	Type	Description
162	UDP	Trap port used to receive SNMP notifications from the nodes. If you want to send traps from the other side of the firewall (e.g. from a Nimbra node) to Nimbra Vision server, this port should be opened.
1099	TCP	Default RMI Registry port. This port is used in the Client-Server communication.
2000	TCP	NMS BE port. This is used in the communication between BE (back-end server) and FE (front-end server). If FE is on the other side of the firewall, this should be kept open. The default Nimbra Vision configuration is to have the FE and BE to run on the same host, meaning that you can normally have this port closed in the firewall.
2001	TCP	NMS FE Secondary Port.

© 2003-2010 Net Insight AB, All Rights Reserved

Port	Type	Description
3306	TCP	For remote access of MySQL database. Only necessary to open if Server Failover is used.
8002	TCP	Client Server communication port. This port is used in Client-Server communication. This port should be opened.
8003	TCP	SAS (SNMP Applet Server) port. In BE - FE combination, all SAS related information is passed through a socket. This port is required only when you give any port number for SAS.
8004	TCP	Config Server port. This port is used in the Client-Server communication.
8005	TCP	Tomcat Shutdown port (For Tomcat 4.0.4 only). You can have this port closed unless you are shutting down Nimbra Vision server remotely. You probably want this port to be closed.
8006	TCP	RMI port. The port on which RMI API opens client and server sockets. This port is used in Client-Server communication.
8009	TCP	Tomcat port. This port is used in Client-Server communication.
9090	TCP	Apache HTTP Server port. This port is used in Client-Server communication, and is also used by the web browser.

Running the Nodes Behind a Firewall

If the network elements are running behind a firewall, the firewall should preferably support FTP connection tracking.

The table below lists the ports that Nimbra Vision uses to access the network elements.

Port	Type	Description
21	TCP	FTP, used for retrieving configurations from the node. FTP will run in passive mode (PASV). This port is accessed from the server.
23	TCP	Telnet, used by the CLI, and for getting configurations from the node. This port is accessed from the server.
80	TCP	HTTP. Used by the element manager (web browser). This port is accessed directly from Nimbra Vision client, and not from Nimbra Vision server.
161	UDP	SNMP, used for most communication to then network elements. This port is used from the server. If the client parameter SNMP_DIRECT is *true* then the Nimbra Vision client will also access the network elements directly. If the parameter is *false* then all access will be relayed via the server. The parameter is configured in `clientparameters.conf`.
>1023	UDP	FTP, used for retrieving configurations from the node. FTP will run in passive mode (PASV). If the firewall supports FTP connection tracking, then is will open the necessary port automatically when needed.

© 2003-2010 Net Insight AB, All Rights Reserved

Node Users

Some functions in Nimbra Vision are using telnet and ftp when accessing the Nimbra network elements. This function allows you to edit the user name and password used in the remote access.

To access the configuration panel, select menu **Tools | Manage Node Users…**.

The Node Users panel contains a list of nodes and user names for remote node access. There is a *default user* is a special user. Its login credentials will be used when an entry has not been added for a node. This user cannot be deleted.

Click **Add…** to add a new entry. This opens the **User Details** dialog where you can enter the user name and password information for a node.

Click **Edit…** edit an already existing entry.

Click **Delete** to delete an entry.

Backup and Restore

Database Backup

When you do a backup of Nimbra Vision database, all the data in the database is saved in a file. You can use this file to restore the whole or only a part of the database at a later time.

You have the following options on how to take a backup:

- Backup when Nimbra Vision server not is running
- Backup when Nimbra Vision server is running

Backup When the Server is Not Running

When Nimbra Vision server is **not** running, backup can be taken using the `BackupDB.sh/bat` file available under the folder *<NMS Home>*`/bin/backup`. By executing the `BackupDB.sh/bat` file, the backup of all the data pertaining to Nimbra Vision is done. By default, the backed up contents are stored under *<NMS Home>*`/backup` folder. The backup filename will bear the date and time of backup. For example, a typical backup filename would look something as follows: `BackUp_JAN1_2003_2_00.data`.

You can specify a different backup folder with the option `-d` *`folder`*. You must specify the folder with a full path name.

Note! *Do not use the backup command when the server is running. When the server is running, it is constantly updating the database tables. Using the script while the server is running would result in inconsistency in the backed up data.*

Backup When the Server Is Running

When Nimbra Vision server is running, backup can be done using a policy. The policy **NMS Backup** takes a backup of the database. You can set up a scheduler for when to take the backups. The backed up contents are stored under *<NMS Home>*`/backup` folder. The backup filename will bear the date and time of backup. For example, a typical backup filename would look something as follows: `BackUp_JAN1_2003_2_00.data`.

© 2003-2010 Net Insight AB, All Rights Reserved

See Also

Restore, to restore a backup.

NMS Backup, policy to make backup.

Restore

Data in the database that has been backed up can be restored using the script `Restore.sh/bat` available under the folder *<NMS Home>*`/bin/backup`. As parameter, give the file name of the backup data file.

For example, if the backup file name is `BackUp_DEC1_2009_2_00.data`, then you restore the contents as (on windows): `RestoreDB BackUp_DEC1_2009_2_00.data`. If the backed up data file is located in some other folder than the default folder, you must supply the full path name for the file, and the path must be entered with forward-slashes, even on windows systems.

The Nimbra Vision server processes must be shut down when doing a restore. Restoring the database will replace all the contents in the database with the restored data.

You can restore a database backed up in an earlier release of Nimbra Vision. After restoring the database, you will have to upgrade the restored database to the format used by the current release of Nimbra Vision. After the database has been restored, navigate to *<NMS_Home>*`\bin\admintools` and run the command script `upgrade.bat`. This will launch the upgrade program. Select "Upgrade current database only".

Note! *Make sure that you type the file name in correct case. The file name is case sensitive.*

See Also

Database Backup

Reinitialize the Database

It is sometimes desirable to start Nimbra Vision server with a fresh database. Occasions when you would like to reinitialize the database are when you have installed a Nimbra Vision server on a host where you have used a previous, incompatible installation. Or you for some other reason would like to have a fresh database.

To reinitialize the database, navigate to *<NMS_Home>*`\bin` using the command prompt or the explorer, and run the command script `reinitialize_nms.bat`. This will drop entire NMS database.

Reinitializing the database will not reset Nimbra Vision server to how it was configured at installation time. Configuration files will not be affected when the database is reinitialized.

© 2003-2010 Net Insight AB, All Rights Reserved

Server Failover

Server Failover

Nimbra Vision server failover is designed to run in environments that require continuous and uninterrupted access to Nimbra Vision. Generally, sudden failure of the server is quite often the reason for interruption of service. A failover mechanism provides a solution for this. At a failover, the redundant or standby server automatically resumes all the functions that were performed by the primary server (i.e. currently active server) upon the failure or sudden termination of the primary server. This topic describes a recommended way to configure Nimbra Vision server failover.

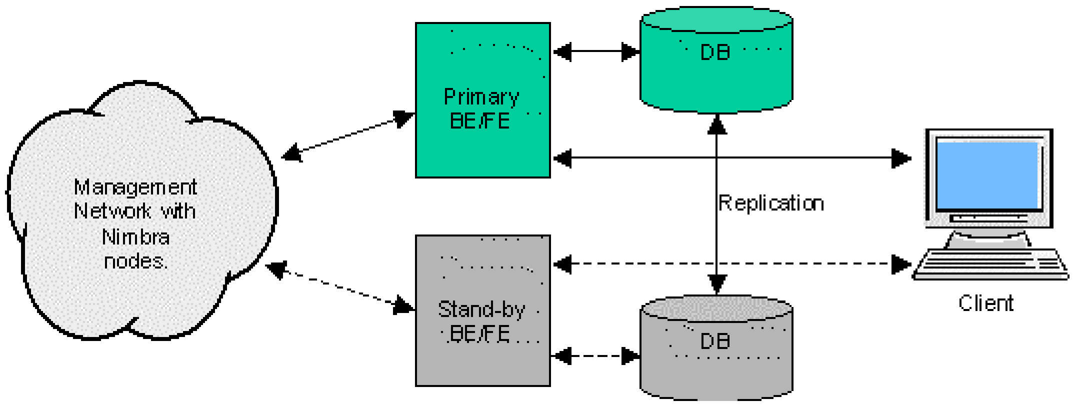
Note! *Nimbra Vision server failover requires additional license key. This license key must be installed before starting the Nimbra Vision server processes.*

Architecture

Nimbra Vision consists of a database server (DB), a back-end server (BE), and a front-end server (FE). The backend server is responsible for updating the database, while the front-end server is responsible for serving the clients. In Nimbra Vision, the back-end server and front-end server are running in the same process, i.e. as a combined BE/FE.

How failover works

To use failover, two servers are used running on different hosts, a primary and a standby server. The two BE/FE servers are started using the same database and they have redundant configurations. When a server starts, it checks the database for status of any other servers. The first server that is started assumes the responsibility as primary server, and registers as such in the database. When the second BE/FE server starts, it detects that a primary server is already started, and assumes the standby role. As a standby server, it is inactive except for monitoring the primary server via the database. The primary (or active) server is periodically increasing a heartbeat counter in the database, and the standby server monitors this counter. If the counter is not increased as expected, the secondary server assumes the role as the primary server. When the failed server is later restarted, it will now assume the role as standby server.



Setup of server failover. Database replication ensures data consistency in the two servers. The client automatically reconnects to active server.

To ensure that the standby server has the same configuration as the primary server, the standby server copies the configuration files from the primary server when the standby server starts, and then periodically, per default every five minutes.

© 2003-2010 Net Insight AB, All Rights Reserved

To have redundancy also on the database, a replicated database is used. This is solved with one database for the primary server and one for the secondary server. Two-way replication ensures that the databases always contain the same information. It is imperative that there are no conflicting updates in the databases from the two servers. Because only the primary server is active and updating the database, there will be no update conflict during normal operation. Nimbra Vision uses MySQL as database.

It is possible to run Nimbra Vision with a remote database. It is important that the primary server and the standby server both are using the same logical database. Note that the scenario explained above is describing one logical database implemented as using two physical database running on two hosts, and that the data replication between the two databases ensures that they hold the same data.

When a user client connects to the server, it connects to the FE server. If the FE server fails, the client will automatically switchover to a redundant FE. When running a combined BE/FE server with failover, the client will automatically connect to the newly activated FE in the new primary BE/FE server.

Setting up Server Failover

Generally, the server will always check for a running database when it starts, and it will start the database only when it fails to detect the running database. (Note that when shutting down Nimbra Vision, it will shut down the database only if it started the database). When using server failover, the database server must be started before starting the standby BE server.

In a typical installation, the Nimbra Vision application is running on the same host as its database, i.e. each Nimbra Vision application is always using the database that is running on the same host as the application itself.

If you are running the database and BE/FE server on the same host, the procedure to setup failover is:

1. Ensure that the hosts file on the standby server contains the same information as on the primary server for all nodes and IP interfaces monitored by Nimbra Vision. (see Setting up the Hosts Database).
2. Install Nimbra Vision on the standby server.
3. Configure both Nimbra Vision servers for use of remote database using the servers own IP address (or hostname) instead of `localhost`. See Remote Database. This is not strictly required when the database is running on the same hosts as the Nimbra Vision application, but recommended because this will make the primary Nimbra Vision to automatically shut down when disconnected from the network.
4. Setup database replication, and keep the database server running (see Database Replication).
5. If you upgrade the license on the server to include Server Failover, then you must install the license key and restart the server.
6. Start Nimbra Vision on the standby server. It will detect that a primary server is already running, and remain in standby mode. It will enter primary mode when it fails to detect presence of the primary server.

You must also setup each node so that it sends SNMP notifications to both servers.

Tuning Server Failover

You can tune the setup of Server Failover. To do this, edit the file *<NMS_Home>*`\conf\FailOver.xml` using a text editor. In this file, you can configure the heartbeat generation interval, the heartbeat monitor interval, the timeout, e-mail notification of failover, configuration file copy interval.

© 2003-2010 Net Insight AB, All Rights Reserved

See Also

Setting up the Host Database

Remote Database

Database Replication

Database

Remote Database

To run Nimbra Vision with a remote database, you must tell Nimbra Vision where to find the remote database. When Nimbra Vision starts, it will try to connect to the database. Only if it fails to find a running database will it start the database by itself.

There are typically two reasons for running or configuring Nimbra Vision for using a remote database:

- Nimbra Vision is using server failover for redundancy with replicated databases
- The database shall be running on a remote for sharing the load.

You need to do the following:

1. To configure Nimbra Vision for a remote database, use a text editor and edit the file *<NMS_Home>*`\conf\database_params.conf`. Default is to connect to the database located at `localhost`. Change the line starting with `url`, replace `localhost` with the hostname or IP address of the database server host. When Nimbra Vision server starts, it will read this file. The following (all in one line) is the default original configuration of host:

```
url
jdbc:mysql://localhost/WebNmsDB?jdbcCompliantTrunc
ation=false  AppModules TopoDB-MapDB-EventDB-
AlertDB-PollDB-PolicyDB-USERSTORAGEDB
```

2. Per default, only root from `localhost` is allowed to connect to the database. You must grant access for the remote user using the GRANT statement in MySQL. See MySQL_Users in Database Administration for details. If you are setting up server failover, and if you are trusting full database access from the redundant server (both ways), then you can grant access to the primary database (before copy) to both servers.

Remote Database for Server Failover

When running redundant Nimbra Vision servers, it is imperative that the two Nimbra Vision servers are never making conflicting database updates. When database replication is used, then the replication will halt if a conflict (or some other error) is detected.

If the Nimbra Vision server is connected to a database on `localhost`, then the connection to the database on `localhost` will remain even when the network connection fails. Assume that the primary host loses network connectivity. It will continue to update the database on `localhost`. The failover host will detect that it has lost contact with the primary server, and will start as new primary, updating its database on `localhost`. We now have two primary servers updating each database. When the network connectivity is restored, the database replication will detect the conflicts and halt the replication.

© 2003-2010 Net Insight AB, All Rights Reserved

To avoid this problem, you should configure Nimbra Vision to use a remote database, even when the database sever is running on the same host as the Nimbra Vision server processes. Configure it to use the hostname or IP address (i.e. not `localhost` nor 127.0.0.1). When the network connectivity fails, the server will loose the connection to the database and will shut down.

You can still run into this problem if the two Nimbra Vision servers lose contact with each other, but not with the network in general.

Remote Database for Load Sharing

To use a remote database for load sharing, you must install MySQL on the remote host. You can copy the `mysql` folder from Nimbra Vision to the remote host. Ensure that Nimbra Vision is not running when copying the database.

See Also

Server Failover

Database Administration

Database Replication

Database Replication

Use of two databases and two-way replication is one method to implements one logical and redundant database.

Database replication is a way of updating a slave database with changes in a master database. MySQL supports one-way asynchronous replication. In Nimbra Vision, we need to setup two-way replication between the two database servers. When running Nimbra Vision with primary and standby servers, each BE server is connected to its database server. This means that both database servers will act as master and as slave at the same time.

When the master database is updated, it writes all updates to binary log files. When the slave connects to the master, it informs the master of the position of its last received update, and the master sends all updates since that position. If there is a disruption in the connectivity between the master and the slave database server, or if the slave database server is shutdown, then when the slave server reconnects, it will be able to continue receive updates from its last position.

Binary logging files that are older than 5 days at log rotation will automatically be deleted. Log rotation happens when a log files reached 100 Mbyte. You can change these settings in file *<NMS_Home>*`\mysql\my.ini`.

This topic discusses:

- Setting up Database Replication
- Verifying Database Replication

Setting up Database Replication

This section describes one way on how to setup database replication for use in Nimbra Vision.

If MySQL binary logging is already enabled, then you do not have to shutdown an already running Nimbra Vision server to setup replication. Binary logging is enabled by default in Nimbra Vision. However, there will be a short disruption in its service when making a snapshot of the database.

In this scenario, we assume two servers, were `host-a` is already running and therefore acting as primary server, and `host-b` is the server that will act as standby server. When setting up database replication, the database servers running on `host-a` and `host-b` will replicate each other's data. Everything

© 2003-2010 Net Insight AB, All Rights Reserved

updated in `host-a` will be replicated to `host-b`, and everything updated in `host-b` will be replicated to `host-a`.

In this section, you will have to edit configuration files using a text editor, and you will have to do some database commands. When editing the configuration file, ensure that you don't make duplicate or contradictive configurations. If you are uncertain about the file format, see MySQL Reference Manual, Using Option Files.

1. Ensure that the two hosts have correct knowledge of the hostname of their masters. E.g. if the name of one of the hosts is `host-a`, then the other host (`host-b` in this example), must use the name `host-a` for the IP address of `host-a`. To check the hostname on a Windows computer, open the *System Properties*, and select the *Computer Name* tab. (You can also use the `hostname` command in a command window). To associate an IP address with a hostname, you can edit the `hosts` database (see Setting up the Host Database).
2. The two database servers must have unique server identities. Also, for replication, binary logging must be enabled. All database configurations are done in the file *<NMS_Home>*`\mysql\my.ini`. Verify the settings using a text editor (e.g. notepad) on `host-a`., The following must be included:

```
[mysqld]
log-bin = mysql-bin
server-id = 1
```

3. If the database server on `host-a` is not started, start the server: run the program *<NMS_Home>*`\mysql\bin\mysqld`. You can start this from the Windows explorer. If Nimbra Vision was already running, and binary logging was disabled, then you have to restart the Nimbra Vision server (actually, it is the database server `mysqld` that must be restarted).
4. Check the text file *<NMS_Home>*`\conf\database_params.conf` for the password that is used by Nimbra Vision to connect to the database. Start a command prompt, and navigate to *<NMS_Home>*`\mysql\bin`, and start the `mysql` client, where *password* is the password:

```
mysql -uroot -ppassword
```

5. The master database server must have an account with at least REPLICATION SLAVE privileges that the slave server can use to connect. Because the two servers will replicate each other, both servers will act as master and as slaves. We create accounts for both masters in the running database on `host-a`. First, create an account that `host-b` can use for connection to `host-a`. If we use username `repl` and password `secret`, execute the statement:

```
mysql> GRANT REPLICATION SLAVE ON *.*
    -> TO 'repl'@'host-b'
    -> IDENTIFIED BY 'secret';
```

   You can replace the hostname with the IP address.

   The semi-colon indicates the end of the command. You can type the complete command on one line. If you do a new line (with e.g. the return-key) before the semi-colon, then mysql client will continue on a new line, which is prefixed with " `->`" as indicated in the example. It will not execute the command until it ended with the semi-colon.

© 2003-2010 Net Insight AB, All Rights Reserved

6. Create an account to be used by `host-a` when accessing `host-b`. For simplicity, we use the same username and password:

```
mysql> GRANT REPLICATION SLAVE ON *.*
    -> TO 'repl'@'host-a'
    -> IDENTIFIED BY 'secret';
```

7. Flush all the tables and block write statements by executing a FLUSH TABLES WITH READ LOCK statement:

```
mysql> FLUSH TABLES WITH READ LOCK;
```

   Leave the client from where you issue the FLUSH TABLES statement running so that the read lock remains in effect. (If you exit the client, the lock is released.) Now, the database cannot be updated. If the Nimbra Vision server is running, it will not be able to update the database. The next operations should therefore be done in haste to minimize the time when the database is locked.

8. While the read lock placed by FLUSH TABLES WITH READ LOCK is in effect, read the value of the current binary log name and offset on the master `host-a`:

```
mysql> SHOW MASTER STATUS;
+------------------+----------+--------------+----
--------------+
| File             | Position | Binlog_Do_DB |
Binlog_Ignore_DB |
+------------------+----------+--------------+----
--------------+
| mysql-bin.000007 |  5600753 |              |
                 |
+------------------+----------+--------------+----
--------------+
1 row in set (0.00 sec)
```

   The File column shows the name of the log and Position shows the offset within the file. In this example, the binary log file is mysql-bin.000007 and the offset is 5600753. Record these values. You need them later when you are setting up the slave on `host-b`. They represent the replication coordinates at which the slave should begin processing new updates from the master.

9. Now, take a snapshot of the database and copy it to `host-b`. You can do this by copying folders *<NMS_Home>*`\mysql\data\mysql` and *<NMS_Home>*`\mysql\data\webnmsdb` to the same location in `host-b`. You could use some archiving program like the *compress to folder* in Windows, or WinZip to zip the files. This might make it easier to copy them to `host-b`.

10. After you have taken the snapshot and recorded the log name and offset, re-enable write activity on the master:

```
mysql> UNLOCK TABLES;
```

11. On `host-b`, ensure that binary logging is enabled and that the server identity is different from the identity on `host-a`; modify the file *<NMS_Home>*`\mysql\my.ini`:

© 2003-2010 Net Insight AB, All Rights Reserved

```
[mysqld]
log-bin = mysql-bin
server-id = 2
```

12. If the database server on `host-a` is not started, start that database server. (If you have followed this guide, the server is already running). To start the server: run the program *<NMS_Home>*`\mysql\bin\mysqld`. You can start this from the Windows Explorer.
13. Start the database server on `host-b`, run the program *<NMS_Home>*`\mysql\bin\mysqld`. You can start this from the Windows Explorer.
14. Start a command prompt on `host-b`, and navigate to *<NMS_Home>*`\mysql\bin`, and start the mysql client.

```
mysql -uroot -ppassword
```

15. In the client on `host-b`, execute the following statement. Replace the options to what is relevant for your system. The values for MASTER_LOG_FILE and MASTER_LOG_POS are taken from what was previously recorded.

```
mysql> CHANGE MASTER TO
    -> MASTER_HOST='host-a',
    -> MASTER_USER='repl',
    -> MASTER_PASSWORD='secret',
    -> MASTER_LOG_FILE='mysql-bin.000007',
    -> MASTER_LOG_POS=5600753;
```

16. Start the slave:

```
mysql> START SLAVE;
```

The database server on `host-b` (slave) should now connect to the to the database server on `host-a` (master) and catch up on any updates that have occurred since the snapshot was taken. We have now finished replication from `host-a` to `host-b`.

17. Now, we must setup replication from `host-b` to `host-a`. In the client on `host-b`, execute the statement SHOW MASTER STATUS and record the File and Position, just as you previously did on `host-a`. Because no other client is connected to the database, it is not being updated (except replication from `host-a`). It is therefore not necessary to lock the database.
18. In the client on `host-a`, execute the CHANGE MASTER statement similar to what you earlier did on `host-b`. Note that we have already setup an account `repl` that allows connections from the master on `host-a`. We did that before copying the database from `host-a` to `host-b`.

```
mysql> CHANGE MASTER TO
    -> MASTER_HOST='host-b',
    -> MASTER_USER='repl',
    -> MASTER_PASSWORD='secret',
    -> MASTER_LOG_FILE='mysql-bin.000001',
    -> MASTER_LOG_POS=98;
```

© 2003-2010 Net Insight AB, All Rights Reserved

19. Start the slave with START SLAVE. Now, the two database servers are replicating each other:

```
mysql> START SLAVE;
```

Verifying Database Replication

If you encounter any problems, check the MySQL log file for any error messages. The log file is named `<NMS_Home>\mysql\data\hostname.err`.

You can run the command SHOW SLAVE STAUS\G to display the replication status. The '\G' indicates end of command, and that each column shall be printed on a new line. The '\G' can be used instead of semi-colon.

```
mysql> SHOW SLAVE STATUS\G
*************************** 1. row ***************************
             Slave_IO_State: Waiting for master to send event
                Master_Host: host-a
                Master_User: root
                Master_Port: 3306
              Connect_Retry: 60
            Master_Log_File: mysql-bin.000004
        Read_Master_Log_Pos: 931
             Relay_Log_File: host-b-relay-bin.000056
              Relay_Log_Pos: 950
      Relay_Master_Log_File: mysql-bin.000004
           Slave_IO_Running: Yes
          Slave_SQL_Running: Yes
            Replicate_Do_DB:
        Replicate_Ignore_DB:
         Replicate_Do_Table:
     Replicate_Ignore_Table:
    Replicate_Wild_Do_Table:
Replicate_Wild_Ignore_Table:
                 Last_Errno: 0
                 Last_Error:
               Skip_Counter: 0
        Exec_Master_Log_Pos: 931
            Relay_Log_Space: 1365
            Until_Condition: None
             Until_Log_File:
              Until_Log_Pos: 0
         Master_SSL_Allowed: No
         Master_SSL_CA_File:
         Master_SSL_CA_Path:
            Master_SSL_Cert:
          Master_SSL_Cipher:
             Master_SSL_Key:
      Seconds_Behind_Master: 0
1 row in set (0.01 sec)
```

Slave_IO_State: indicates the current status of the slave. When replication is running, this message is `Waiting for master to send event.`

Master_Host: the name or IP address of the slave host. This is the host you provided with the CHANGE MASTER statement when setting up replication.

© 2003-2010 Net Insight AB, All Rights Reserved

Master_Log_File: the name of the binary log file on the master server. Initially, this is the name you provided with the CHANGE MASTER statement when setting up replication, but this is updated when new binary log files are used.

Slave_IO_Running: shows whether the IO thread for the reading the master's binary log is running. This is `Yes` if the replication is running.

Slave_SQL_Running: shows if the slave SQL slave thread is running. This should be `Yes`. If a statement on a slave produces an error, the slave SQL thread terminates, and the slave writes a message to its error log. This would typically happen if the same data were modified on the slave and on the master. You should then connect to the slave manually and determine the cause of the problem. When you have fixed the problem, issue the START SLAVE statement to start the replication again.

Note! *If Nimbra Vision server has been running as primary servers at on both hosts at the same time, then the two servers modifies the same data records in both databases, and the replication stops. The recommended solution to fix this is to set up replication again using one (the best) of the databases as initial master.*

Last_Error: shows the last error registered when processing the relay log. Ideally this should be blank, indicating no errors.

Seconds_Behind_Master: shows the number of seconds that the slave SQL thread is behind processing the master binary log. A high number (or an increasing one) can indicate that the slave is unable to cope with the large number of queries from the master.

See Also

MySQL Reference Manual, Using Option Files (at www.mysql.com)

MySQL Reference Manual, Chapter 6 Replication (at www.mysql.com)

Database Administration

This chapter shortly describes some of the commands and actions when administrating the MySQL database in Nimbra Vision. The following id described:

- Starting MySQL server
- Stopping MySQL server
- Installing as MySQL as Windows Service
- Removing MSQL as Windows Service
- MySQL Users
- User name and password used by Nimbra Vision for MySQL

Starting and Stopping the MySQL Server

Starting the Server

The database server is automatically started when starting the Nimbra Vision server. However, when using server failover, you must start the database before starting the standby server. To start the database server, navigate to `<`*`NMS_Home`*`>\mysql\bin` and start `mysqld`.

© 2003-2010 Net Insight AB, All Rights Reserved

Stopping the Server

If the Nimbra Vision server started the database server when it was starting, then it will be automatically shutdown when the Nimbra Vision is being shutdown.

If the database server was started before the Nimbra Vision server, then you must shut down the database server manually. Open a command prompt and navigate to `<NMS_Home>\mysql\bin`, and run:

```
mysqladmin -uroot -ppassword shutdown
```

This assumes the password *password*. The password is defined in the file `<NMS_Home>\conf\database_parameters.conf`.

Installing and Removing MySQL as a Windows Service

Installing MySQL as a Windows Service

If you have installed Nimbra Vision as a Windows Service (see Windows Service in the Installation Manual), and you are using server failover, then you must install the database server as a service as well. Installing the service does not start the service, but the service will automatically be started at boot. To install, navigate to `<NMS_Home>\mysql\bin` and run:

```
mysqld --install
```

To start the service, run:

```
NET START MySQL
```

Removing MySQL as Windows Service

To uninstall or remove MySQL as a service, you must first stop the running service. After it has been stopped, you can remove it. Open a command prompt and navigate to `<NMS_Home>\mysql\bin`. Run:

```
NET STOP MySQL
mysqld --remove
```

MySQL Users

To execute MySQL statements, open then mysql client: Start a command prompt and navigate to `<NMS_Home>\mysql\bin`, and start the `mysql` client. The password is available in `<NMS_Home>\conf\database_params.conf`.

```
mysql -uroot -ppassword
```

Adding a User

To add a user with privileges, you can use the GRANT statement. The example below grants all global privileges to the user *username* that can log in from node *hostname* using password *password*:

```
mysql> GRANT ALL ON *.*
    -> TO 'username'@'hostname'
    -> IDENTIFIED BY 'password';
```

© 2003-2010 Net Insight AB, All Rights Reserved

Removing a User

Use DROP USER statement to remove a user. The following example removes the user *user* that can login from host *hostname*'.

```
mysql> DROP USER 'user'@'hostname';
```

List Users

Use the SELECT statement to list users. The following example list the username and host the user can log in from:

```
mysql> SELECT USER,HOST FROM MYSQL.USER;
+------+--------------+
| USER | HOST         |
+------+--------------+
| repl | 10.100.3.129 |
| repl | 10.100.3.165 |
| root | localhost    |
+------+--------------+
3 rows in set (0.00 sec)
```

User name and password used by Nimbra Vision for MySQL

The user name password that Nimbra Vision uses when connection to t he database is defined in the file *<NMS_Home>*`\conf\database_parameters.conf`.

Configuration Parameters

Setting Link Thickness in Map

You can configure how links shall be presented in the DTM map. The width depends on the transmit capacity of the DTM interface where the DTM link originates. Configuration is done by means of the configuration file *<NMS_Home>*`\conf\serverparameters.conf`.

When link is added to a map, Nimbra Vision will assign the link symbol in the DTM map with a thickness depending on the capacity of the link (or, actually, the transmit capacity of the originating interface). It will also assign the property controlling whether the label shall be displayed or not.

Parameter	Description
CapacityLimits	This is a comma-separated list with capacity limits given in number of 512 kbit/s slots. Together with the **LinkThicknesses**, it will describe the thickness of the link for different capacities. The list shall include the limits in an increasing order. For each value in the list, the link will have the thickness denoted by the value in the same position in the **LinkThicknesses** list.

© 2003-2010 Net Insight AB, All Rights Reserved

Parameter	Description
LinkThicknesses	This is a comma-separated list with thicknesses. The first value in the list is the thickness to use for links with capacity up to and including the first value of **CacapityLimits**. The second value is the thickness to use for links exceeding the first but not the second value of the **CapacityLimits**, and so on. You can configure as many intervals as you find necessary. The values in the list are non-negative integers. A value of 0 indicates that a thin dashed line shall be used for painting the link. Values greater than 0 indicate the line thickness in pixels. Note that you can also directly change the value of the link symbol property **thickness** to control how to display individual links.

***Note!** There is one item more in the list **LinkThicknesses** than in the **CapacityLimits**. The last **LinkThicknesses** value will be used for all links with capacity exceeding the last entry in **CapacityLimits**.*

Example of a file is shown below. It will display links with capacity up to and including 65 slots as a thin dashed line, links capacity from 66 slots up to and including 1790 slots as one pixel thickness, links with capacity from 1791 up to and including 1940 slots with two pixel width, and links exceeding 1940 slots with three pixel width.

```
…
CapacityLimits 65,1790,1940
LineThicknesses 0,1,2,3
…
```

Configure Alarm Printer

Alarm and events can be printed form the Alarm or Event panels using the menu **Action | Print**. Alarms and events will be printed as plain text file to a printer that is configured on the server.

To configure a printer, you have to edit the file *<NMS_Home>*`\conf\NmsProcessesBE.conf`. Find the entry `#java com.adventnet.nms.eventdb.EventMgr` in the file. Append the following argument to the list of arguments configure the printer:

```
PRINT_COMMAND "print /d:\\\\printserver\\printer .\\state\\printfile.tmp"
```

This assumes that the printer is a network printer called `printer` on the print server called `printserver`, i.e. the printer is `\\printserver\printer`.

© 2003-2010 Net Insight AB, All Rights Reserved

Configure Software/Firmware Upload

You can change the behavior of the Software/Firmware Upload functions with parameter settings in the file `<NMS_Home>\conf\serverparameters.conf`.

Parameter	Default	Description
MAX_CONCURRENT_UPLOADS	30	The maximum numbers of concurrent upload sessions. Nimbra Vision will open up to this number of sessions, and start the upload process for this number of network elements in parallel. Upload to remaining network elements are queued, and will start when a running upload session has terminated.
NODE_UPLOAD_TELNET_TIMEOUT	900	Number of seconds before timeout. Nimbra Vision closes the telnet connection to the network element after the specified period if no data has been received from the network element.
NV_CMD_UPLOAD		The shell command to run on the network elements to upload software/firmware. The result shall present progress in the format `n/m`, where `n` is the current step and `m` is the total number of steps. The token `$URL$` will be substituted with the provided URL.
NV_CMD_RESTART		The shell command to run on the network element to restart units.

Other Configuration

Setting Initial Serial Number for Services

Each service that is created from using Nimbra Vision is assigned a serial number. The serial number is a number that is incremented by one for each new service. Nimbra Vision maintains the next serial number in its database. Whenever Nimbra Vision is discovering a new service, it compares its serial number with nest serial number. If the serial number of the discovered service is greater than the next serial number, then the maintained value is updated with the serial number from the discovered service. This ensures that the maintained value is automatically updated with what is actually used in the network.

© 2003-2010 Net Insight AB, All Rights Reserved

Use the command file *<NMS_Home>*`\bin\admintools\ServiceSerialNumber.bat` to set or view the serial number for the next created service.

To set the serial number, open a command prompt, navigate to the folder *<NMS_Home>*`\bin\admintools` and run the command `ServiceSerialNumber.bat` with the serial number as parameter. The serial number is a natural number.

To view the next serial number, run the command without any parameters.

You must know the database password. The password is specified in the file *<NMS_Home>*`\conf\database_parameters.conf`.

Recommended Nimbra Vision Configuration

After Nimbra Vision has been installed and setup, Nimbra Vision can be used. This topic describes some additional settings or configurations that should be considered. Some of these tasks can be configured to be run periodically.

- **Defining personal user accounts** with permissions depending on roles. Each user should have an individual user logins. It is possible to define groups representing different roles. A user can be a member of different groups, thus getting the permissions defined for each role. See Security Management.
- **Update hostnames in nodes**. When using Service Provisioning, then all nodes must know the hostnames. Create a HostListPush policy to keep the hostnames list in the nodes updated. See Host List Push.
- **Purging Archived Events** from the database. Events are stored in a dedicated table in the database archival. Old events should be removed to prevent it from filling the database. See Cleanup Archived Events.
- **Purging archived PM reports** from the database. G.826 performance reports are stored in a dedicated table in the database for archival. Old performance reports should be removed to prevent it from filling the database. See Cleanup Archived PM Reports.
- **Purging old statistical data** from the database. Statistical data is saved to in the database. Bt default, the usage of the trunk interfaces, and some packet counters on IP interfaces are saved as statistical data. This data is used when creating reports. Old statistical data should periodically be removed from the database to prevent filling the database. See Statistics Table Cleanup for definition of a policy that removes old data. Also, see Introduction to Performance Monitoring.
- **Backing up Nimbra Vision database**. Nimbra Vision relies on a SQL database for all its data. This database should periodically be backed up. If the server is ungracefully shut down, the database may get corrupt due to cached data that is lost. Database is preferably backed up using a policy. See Backup.
- **Purging old NMS database backups** from the file system. If backups of Nimbra Vision database are made, it is necessary to ensure that the backup files do not fill up the file system. Database backups are stored in the folder *<NMS_Home>*`\backup`, one file for each backup. Removal of files in this folder must be done from the operating system.
- **Backing up the nodes' configurations**. The node's running configuration, i.e. how the node id currently configured, can be saved locally on the node. A policy can be setup to periodically save this running configuration on a set of nodes. This policy can also make a

© 2003-2010 Net Insight AB, All Rights Reserved

copy of the configuration to Nimbra Vision server for e.g. backup purposes. See Save DTM Node Registry.

- **Purging old node configuration backups** from the file system. If a policy is setup to copy the nodes configuration to Nimbra Vision server, it is necessary to ensure that these backup files do not fill up the file system. The default location where the files are stored is in the folder *<NMS_Home>*`\backup\nodeconfigurations.` Removal of files in this folder must be done from the operating system.
- **Generate daily, weekly and/or monthly reports**. Policies can be setup to generated HTML reports with summaries of trunk interface usage. Multiple policies can be setup to generate reports daily, weekly or monthly. See Generate Reports.
- **Purging old reports** from the file system. Generated HTML reports with statistical information abut the trunk interfaces are stored in the folder *<NMS_Home>*`\reports\dtmIfReports` on Nimbra Vision server. It is necessary to ensure that this folder does not fill up the file system. It is safe to just remove the files representing the reports (possibly the oldest files). When new reports are generated by a policy, the index pages are automatically updated with the reports remaining in the folder. It is therefore recommended to remove old files just before new reports are generated. Removal of files in this folder must be done from the operating system.

© 2003-2010 Net Insight AB, All Rights Reserved

Appendices

Setting up SSH Tunneling for Use with Nimbra Vision Client

Nimbra Vision uses RMI between the client and the server. RMI is a method for a client to access an object on a server. When the client does RMI lookup, the server generates a stub and hardcode its own address in it. The stub is then transferred to the client and is then used by the client to access the object on the server. Because the address is hardcoded in the stub, the client will attempt connect to whatever address that is hardcoded. This works fine as long as the client and server use the same address for the server. But if the connection to the server is made through an SSH tunnel (or if the server is behind a NAT firewall), then the client would use a different address for the server; the client would use 127.0.0.1 if SSH tunneling with port forwarding is done from the client. You can configure the Nimbra Vision server to associate a different address in its stubs. You need to specify an address, or hostname, that the client can use to access the server. Note that if you need to be able to connect clients both using and not using SSH tunneling, then you must specify a hostname that would resolve to correct IP address for both types of clients.

In this text, we assume that the hostname of the Nimbra Vision server is `nv-server`.

On the client, the hostname (e.g. `nv-server`) shall resolve to the IP address that the Nimbra Vision client shall use to access the server. When SSH tunneling is used from the client host, then it should resolve to 127.0.0.1.

To specify that hostname `nv-server` shall resolve to 127.0.0.1, edit the hosts file located in the folder `C:\WINDOWS\system32\drivers\etc`. Append the hostname (e.g. `nv-server`) to the entry for 127.0.0.1, as:

```
127.0.0.1    localhost nv-server
```

Entries in the hosts file will override DNS.

To specify the hostname that RMI will associate with the remote stubs, you must set the JVM property `java.rmi.server.hostname`, which is assigning it the hostname (or address):

```
-Djava.rmi.server.hostname=<hostname>
```

To do this, edit the server startup file `bin\startnms.bat`; add the parameter to the line that is just after the line ":start1", which then becomes:

```
%JAVA_HOME%\bin\java -Djava.rmi.server.hostname=nv-server \
-Xbootclasspath/p:customclasses/nvrt.jar...
```

© 2003-2010 Net Insight AB, All Rights Reserved

(Note: The backslash (\) denotes line continuation. An actual command must appear within a single line.)

If you have installed and shall be running the Nimbra Vision server as a service, then you must edit the file java_service.ini that has been installed in the windows folder (`C:\WINDOWS`). Add the parameter to the beginning of the `SERVER_JAVA_OPT` definition, which becomes:

```
SERVER_JAVA_OPT=-Djava.rmi.server.hostname=nv-server \
-Dresource_check=1099,2000,2001,8002,8003,8004,8009,8005...
```

(Note: The backslash (\) denotes line continuation. An actual command must appear within a single line.)

Setting up SSH tunneling on the server

Install and configure an SSH server on the server. An SSH server that has been tested is copssh (http://www.itefix.no/i2/copssh), which is a OpenSSH distribution for Windows. This server is simple to install and configure.

Setting up SSH tunneling on the client

When setting up SSH tunneling, all the ports used by the Nimbra Vision client to access the Nimbra Vision server must be forward to the remote host's external IP address (or a hostname that the server would resolve to its external IP address). Assuming that the hostname of the server is `nv-server`, the port forwarding configuration becomes:

Source Port	Destination
1099	nv-server:1099
2000	nv-server:2000
2001	nv-server:2001
8002	nv-server:8002
8003	nv-server:8003
8004	nv-server:8004
8006	nv-server:8006
8009	nv-server:8009
9090	nv-server:9090

The source ports are local ports.

A good SSH client for Windows is Putty (http://www.chiark.greenend.org.uk/~sgtatham/putty). To configure Putty for SSH tunneling, do the following:

1. On the **Connection | SSH | Tunnels** panel, configure the port forwarding. For each port as specified above, add Source port and Destination and ensure that the radio buttons Local and Auto are selected, and click the Add button.
2. On the Session panel, enter IP address of the Nimbra Vision server and select connection type SSH. Also, give the session a name in Saved Sessions, and save the session by clicking the Save button.

© 2003-2010 Net Insight AB, All Rights Reserved

3. Now you can open a session and login to the server.

Copssh also comes with an ssh client.

Used Software

Putty 0.60

Copssh 2.1.0

Partitioning the Nimbra Network

The use of tags on nodes in combination with the security settings (Custom View Scopes, Operations and Scopes) is a powerful tool for partitioning the Nimbra network into different domains. These domains can be used to control access to only part of the network for some organizations. Different organizations can access different parts of the network. This topic discusses some examples on how to use tags and to setup permissions to accomplish some different policies.

The examples in this topic are not the only way to accomplish the policies.

The following policies are discussed:

- Grant access only to the organization's nodes
- Grant service provisioning only between the organization's nodes
- Control services' paths through the network for an organization
- Allow one organization to share a service with another organization

Grant access only to the organization's nodes

1. Define a tag for the organization, i.e. choose a word that shall be used as tag representing the organization.
2. Set the tag on all the nodes and recursively on all sub-objects on the node (interfaces and services). All objects later created on these nodes, and all events and alarms on these nodes will become tagged with the chosen tag.
3. Create a group that the users in the organization shall be member of. This group will define the permissions for the users. You can use the default groups as an example on how to organize operations for different roles into different groups.
4. Assign Operations (set permissions) to the group. This will grant the group to perform the operations that are allowed.
5. Set the users that shall be a member of the group.
6. Set up the permitted operations for the group by setting Scopes for the operations. The scopes will limit grant of the operation to only objects matching the criteria. Set the criterion to include property named **tag** with the value as the defined tag. (No wildcards etc. allowed here!). For services, or at least of the `Create Service` operation, you should also include property named **destinationTag** with the value of the defined tag to enforce that the services must terminate on the organization's nodes. Note that you can include `NULL` in the comma-separated list of values to indicate that the service can have no destinations.
7. Set up the Custom View Scopes for the group. Setup Authorized Scopes for all the Custom View Scopes (Events, Network Database, Alerts, Maps, Stats Admin). For each Custom View Scope, setup the Authorized Scopes

© 2003-2010 Net Insight AB, All Rights Reserved

with match criteria to include property named **tag** with value as the defined tag. You can further limit the match to exclude (or only include) objects matching some other criteria as well, e.g. exclude `Interface` manage objects, or include only events/alarms on services.

8. Assign the Authorized Scopes to the group. Each Custom View Scope will filter objects in the database views to only include the tagged objects for the users in the group.

***Note!** A user can be a member of multiple groups, and will be granted permission of all the groups.*

Grant service provisioning only between the organization's nodes

Tags can be used for controlling how an organization can setup services in the network.

1. Define a tag that shall be for the organization's use, a private tag. The purpose with this tag is to mark objects such as nodes, interfaces, links and services that belong to the organization.
2. Set the tag on all nodes that belongs to the organization. New services will inherit the property **tag** from the originating node, and services will always use the tag of the destination node for the property **destinationTag**.
3. Create a group that represents the role of service provisioning for the organization. This group shall be assigned to all the users in the organization.
4. For this group, set up permitted operations to allow operations `Create Service`, `Get Service`, `Modify Service`, `Delete Service` and `Use Source Route.`
5. For the operations `Create Service` and `Modify Service`, add a scope with property named **tag** set to the value of the private tag, and **destinationTag** set to the value of the private tag and `NULL` separated by a comma. This will allow the organization to create and modify service that originate in its own nodes, and that does not have a destination or with a destination in its own nodes.
6. For the operations `Get Service` and `Delete Service`, add a scope with property named **tag** set to the value of the private tag. This will allow the organization to view and delete any service originating in its own nodes.
7. The operation `Use Source Route` is necessary to be able to setup a service.

Control services' paths through the network for an organization

By creating source routes, and specify exactly what source routes the organization must use, you can restrict the organization to only these source routes.

1. Create source routes in all the nodes. The source routes shall describe all the allowed paths. Smart naming of the source routes can make it simpler to setup the scopes later. If the same names are used for source routes on

© 2003-2010 Net Insight AB, All Rights Reserved

all the nodes, then the scope does not have to include then node's name in its criteria.

2. Set up the permitted operations for the group to not allow the operation `Modify Source Route`, and to allow the operation `Use Source Route`.
3. If the organization shall be allowed to use any of the defined source routes (including to not use a source route at all), then nothing more needs to be setup. This is the most common situation when an organization has its own nodes. But if the organization shall be allowed to use only some specific source routes, then these must be listed for each node in scopes. To do that, set a Scope for the operation `User Source Route`. The scope shall have matching criteria with property named **routeName** set to a comma-separated list of the names of the allowed source routes. If the use of no source route shall be allowed, include the route name `--none--`. If different nodes have different named source routes, then create multiple scopes with the properties **name** and **routeName**, one for each node. (If some nodes have source routes named the same, then you can list the nodes and the route names in the same scope).

Allow one organization to share a service with another organization

Sometimes an organization wants to share a service with another organization. One such example is a content provider that shares a data stream (with e.g. video) to another organization. In this case, the content provider is the owner of the service, and should have exclusive full control of the origination of the service. The receiving organization should not be allowed to modify the service, this should only be allowed by the content provider. The receiving organization should be able to connect to the shared service, and should have exclusive control of the destination of the service. The content providing organization should not be allowed to alter e.g. the terminating interface.

For the content sharing organization, do:

1. Define a tag that shall be used for shared services. The purpose with this tag is to mark services that the content providing organization shares with the content receiving organization.
2. Create a group that represents the role to share services. This group shall be assigned to the users in the content providing organization.
3. For this group, set up permitted operations to allow operations `Create Service`, `Get Service`, `Modify Service`, `Delete Service` and `Set Tag`.
4. For the operation `Create Service`, add a scope with property named **tag** and value set to the shared tag, and property **destinationTag** set to the content providing organizations private tag. This will enable the content provider to add its own destinations to its shared services. This is step is otherwise not necessary.
5. For the operations `Get Service` and `Delete Service`, add a scope with property named **tag** and value set to the shared tag. This will allow the content providing organization to view the service in the service editor, and to delete destinations (regardless of if any current destination has been added by the receiving organization). The content providing organization shall not be allowed to modify the service using the Service Editor when

© 2003-2010 Net Insight AB, All Rights Reserved

the service has the shared tag set because this may include setting of the destination's interface.

6. For the operations `Modify Service`, add a scope with property named **tag** and value set to the shared tag, and property **destinationTag** set to `NULL`. This will allow the content providing organization enable/disable the service from a selected service in the Network Database, or to otherwise modify the service using the Service Editor when the service is not used by a service receiving organization. The content providing organization shall not be allowed to modify the service using the Service Editor when the service has the shared tag set because this may include setting of the destination's interface.
7. For the operation `Set Tag`, add a Scope with the property name set to **tag**, and the value a comma-separated list of the organization's tag and the share tag, and a property named **classname** set to `DtmService`. This will allow the organization to change the tag between the shared and non-shared tag on services.
8. For the group, set up an Authorized View for the custom scope `Network Database`. The authorized scope shall have matching criteria with the property named **classname** and value `DtmService`, and tag with the value of the share tag. This will allow the organization to view the shared services in custom views.

For the content receiving organization, do:

1. Create a group that represents the role to receive shared services. This group shall be assigned to the users in the receiving organization.
2. For this group, set up permitted operations to allow `Create Service`, `Get Service` and `Delete Service`. The group shall not be allowed the operation `Modify Service`, only the content provides shall be able to do that.
3. For the operation `Get Service`, add a scope with property named **tag** set to the shared tag. This will allow the receiving organization to view the shared service in the Service Editor.
4. For the operations `Create Service` and `Delete Service`, add a scope with property named **tag** set to the shared tag, and **destinationTag** set to receiving organization's private tag and to `NULL` separated by a comma. This will allow the organization to add and delete destinations from the shared service, but only if it terminates in the organization's own nodes.
5. For the group, set up an Authorized View for the custom scope `Network Database`. The authorized scope shall have matching criteria with the property named **classname** and value `DtmService`, and tag with the value of the share tag. This will allow the receiving organization to view the shared services in custom views.

When the service in not shared, the content provider:

- Can share the service by setting the tag to the shared tag.
- Can display the service in custom views.
- Can modify the service using the Service Editor, but only if no content receiver is using the shared service.
- Can remove any destination.

© 2003-2010 Net Insight AB, All Rights Reserved

- Can delete the complete service.

When the service is not shared, the content receiver:

- Cannot display the service in the custom views.
- Cannot access the service.

When the service is shared, the content provider:

- Can unshared the service by setting the tag to the content provider's own tag.
- Can display the service in custom views.
- Cannot modify the service using the Service Editor; this is to prevent the content provider to modify the content receiver's selection of interface.
- Can enable or disable the service from the Network Database (custom views).
- Cannot add a destination (to another organization's nodes).
- Can remove any destination.
- Can delete the complete service.

When the service is shared, the content receiver:

- Cannot control whether a service is shared or unshared.
- Can display the service in custom views.
- Cannot modify the service using the Service Editor, this is to prevent the content received to modify the content provider's selection of interface, or any other setting such as capacity.
- Cannot enable or disable the service from the Network Database (custom view)
- Can add a destination to its own nodes, and at that time select the destination interface.
- Can remove a destination that is terminating in its own nodes.
- Can delete the complete service if it has no destination, or if all its destinations terminate in its own nodes.

When a service is created, it will inherit the tag from the node where it is originating. This is the mechanism that prevents the content receiving organization from creating a service with the share tag on the content provider organization's nodes. When a destination is added to an existing service, then the tag has already been set on the service. This is what will allow the content receiving organization to add a destination to the shared service.

© 2003-2010 Net Insight AB, All Rights Reserved

Glossary

1

1+1 protected: A connection that is using two channels through the network is 1+1 protected. One channel is active, the other channel is in stand-by. Data is transmitted on both the channels, and the receiver is deciding which channel to use. 1+1 protected channels have very short fail-over time.

A

admin status: The desired state of a resource, as configured.

C

channel: A channel is a set of DTM slots reserved from an originating to a terminating entity. It is used for sending data. A channel can be switched through different nodes, and although the number of slots is the same for the channel through the network, the actual slot allocation is different between each node.

channel id: The cannel id is an identifier for a channel. The identifier is a number assigned by the node where the channel originates. It is unique within the originating node. The channel id together with an identity of the originating node uniquely identifies a channel in the entire network.

connection: The virtual circuit set-up for a service between two trail termination points. The connection can use one or two channels depending on whether it is 1+1 protected or not.

control channel: A channel between adjacent network elements used be the network for control data.

D

DLE: DTM LAN Emulation, an internal service used for setting up IP segments on top of the DTM network. The DLE segments are parts of the in-band management network.

DSTI: DTM Service Type Instance, a within the service type, unique number that identifies the instance of the trail termination point. The DSTI is used to identify the TTP within a remote network element.

DTM interface: A DTM interface is a logical interface connecting nodes in a DTM network. The logical interface holds a number of DTM slots, i.e. a capacity. It is independent on the underlying physical network characteristics, and the only difference between different types of interfaces is the available slot range, i.e. the capacity of the interface. The DTM interface is always depending on an underlying trunk interface, e.g. different type of SDH interfaces. Typically, there is exactly one DTM interface for exactly one trunk interface, meaning that a DTM interface corresponds to a trunk interface.

© 2003-2010 Net Insight AB, All Rights Reserved

DTM link: A DTM link is a logical connection between two neighboring nodes' DTM interfaces, i.e. it describes how the neighboring nodes are logically connected. When exactly one DTM interface corresponds to exactly one trunk interface, then the DTM link corresponds to the physical connection between the trunk interfaces. A DTM links can be uni-directional or bi-directional, but requires two way communication between the neigboring nodes, i.e. the return link may be another DTM link.

DTM network map: A map that graphically displays the network elements and the DTM by-pass link connections.

E

ETS: Ethernet Transport Service. Service to transport Ethernet packets. It offers a transparent wide-area layer-2 transport service for Ethernet traffic fully compliant with IEEE 802.3. The Ethernet Transport Service is an end-to-end service, or rather edge-to-edge service. It creates point-to-point (unicast) or point-to-multipoint (multicast) connections over the fiber network that is spanned by the network elements.

ETSv1: Ethernet Transport Service version 1. The configuration model used for Ethernet services up to NimOS GX4.4.

ETSv2: Ethernet Transport Service version 2. The configuration model used for Ethernet services from NimOS GX4.5. The functionality of ETSv2 is compatible with ETSv1, but the configuration model is different. ETSv1 and ETSv2 can co-exist in the network.

I

in-band management network: An IP network emulated in the DTM network that is used for network management of the network elements. The network does not require any external equipment, except one entry-point.

ITS: Isochronous Transport Service. A service to transport services such as ASI, SDI, HD-SDI, AES/EBU and PDH .It offers a transparent wide-area layer-2 transport service for the type of traffic. The Isochronous Transport Service is an end-to-end service, or rather edge-to-edge service. It creates point-to-point (unicast) or point-to-multipoint (multicast) connections over the fiber network that is spanned by the network elements.

M

managed: A managed object is considered managed if the NMS shall monitor its status.

managed object: A representation of the actual network element, interface, link or other device that shall be managed by the NMS.

managed objects: A representation of the actual network elements, interfaces, links or other devices that shall be managed by the NMS.

N

network database: The database containing all the managed objects. Different views can be applied on the database to display a sub-set of managed objects, with a defined set of properties.

O

oper status: The operational state of a resource. This is the actual state that the resource has. It normally follows the admin status.

© 2003-2010 Net Insight AB, All Rights Reserved

out-band management network: An IP network for managing the network elements. The management network is external to the DTM network, and requires that all network elements are connected to the IP network.

R

redundant networks: The NMS management station and the network elements are connected to multiple IP networks, where the IP networks are independent of each other. If one IP network would fail, the management station would still be able to reach the network elements using the other network. The networks are redundant.

S

service interface: The physical interface for connecting the external equipment from/to where data is transported, e.g. an ASI, PDH or Ethernet interface.

services: A service is the transport that the network is providing. This could be transport of e.g. SDI, ASI, PDH, SDH, or transport of Ethernet data.

slot: A slot is the smallest unit of data that can be allocated. A slot is always 512 kbit/s. A channel allocates the number of slots that is required for the channel capacity.

statistics: The definition of collected data. The statistics e.g. what data to collect and how often to collect.

T

Tree: The tree is present on the left-side of the Application Client and displays set of hierarchical data. The fundamental object in a tree is called a node, which represents a data item in the given hierarchical set. Thus, a tree is composed of one or more nodes. By choosing a particular node, the corresponding panel is displayed on the right-side display panel.

trunk interface: The physical interface to connect the trunks that interconnects the DTM network elements. This is the high-speed interface where the DTM by-pass link connections are set-up.

TTP: Trail Termination Point. A logical entity in the network element between which connections are established. The TTP also connects to the service interface, which completes the end-to-end transport.

U

unmanaged: A managed object is considered unmanaged if the NMS shall not monitor its status.

© 2003-2010 Net Insight AB, All Rights Reserved

Index

© 2003-2010 Net Insight AB, All Rights Reserved

© 2003-2010 Net Insight AB, All Rights Reserved

© 2003-2010 Net Insight AB, All Rights Reserved