VMware Smart Assurance Service Assurance Manager Adapter Platform User Guide

VMware Smart Assurance 10.1.0

You can find the most up-to-date technical documentation on the VMware website at:

https://docs.vmware.com/

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

Contents

Preface

1

As part of an effort to improve its product lines, VMware periodically releases revisions of its software and hardware. Therefore, some functions described in this document might not be supported by all versions of the software or hardware currently in use. The product release notes provide the most up-to-date information on product features.

Contact your VMware technical support professional if a product does not function properly or does not function as described in this document.

Note This document was accurate at publication time. Go to VMware Online Support docs.vmware.com to ensure that you are using the latest version of this document.

Read the following topics next:

Purpose

This guide is intended for administrators who are responsible for deploying, installing, and configuring the Adapter Platform. IT managers who seek to understand the role of the Adapter Platform and the Global Manager in the context of an solution may also find this guide useful.

In addition to the configuration guides for specific components, administrators should also read the Service Assurance Manager Deployment Guide and the System Administration Guide.

Audience

This document is intended for administrators who are responsible for deploying, installing, and configuring the Adapter Platform.

Service Assurance Adapter Platform

2

Read the following topics next:

Overview

The Service Assurance Manager Adapter Platform (Adapter Platform) imports and normalizes topology and event information from sources other than products, such as SNMP traps, system log files, or events generated from the **sm_ems** command line interface. After it normalizes to the ICIM data model, the information is transferred to the Service Assurance Manager (Global Manager).

The Adapter Platform works with third-party applications in a variety of ways to import topology and event information and prepare it for use by the Global Manager. The Adapter Platform allows for several methods of receiving events, including:

- SNMP trap integration
- Log file integration
- Integration by using a custom adapter
- Command line interface

  The Adapter Platform works with any combination of these methods.

  Regardless of how the Adapter Platform receives the event information, it does the following:
- Provides uniform representation of event information regardless of the source of the event. This is called event normalization.

- Consolidates recurring events, rather than considering each new event separately. This reduces the number of events sent to the Global Manager. It is called deduplication.
- Allows for an incoming message to clear a previous event.
- Provides a mechanism to set an event expiration. This mechanism automatically clears events that have not changed for a period of time.
- Optionally associates events to a topology element, thus placing events in their topological context.

  After the event information is processed, it is sent to the Global Manager.

  Adapter Platform configuration files describes the configuration files that are relevant to the Adapter Platform Server. The **sm_edit** utility ensures that modified files are saved to the appropriate location in the *BASEDIR*/smarts/local directory.

  Note You should edit only local copies of the Adapter Platform configuration files, which are located in *BASEDIR*/smarts/local/conf/icoi. If local copies of the configuration files already exist and you attempt to edit non-local files, the **sm_edit** utility will locate and edit the local (not the non-local) versions of files. The sm_edit utility provides information about the **sm_edit** utility. System Administration Guide provides detailed instructions about how to properly modify an file.

Table 2-1. Adapter Platform configuration files

Directory under *BASEDIR*	Filename	User editable	Description
smarts/conf/icoi	bootstrap.conf	No	Contains vital information required to start the Adapter Platform Server.
smarts/conf/icoi	dxa-sysip.conf	No	Used by the Adapter Platform to import information from cooperating IP Availability Managers.
smarts/conf/icoi	icoi-config.dtd	No	Definition file for XML configuration.
smarts/conf/icoi	icoi-default.xml	Yes	Contains the default notification lists, user profiles, and users for the Service Assurance Adapter Platform when the server is first started, or started without an existing repository file.
smarts/conf/icoi	icoi-config-sample.xml	Yes	Sample XML file of notification lists, user profiles, and users. It can be used as a template for creating XML files of these entities.
smarts/conf/icoi	nlconfig-sample.xml	Yes	Sample XML file for notification list configuration. It can be used as a template for creating additional files.

Table 2-1. Adapter Platform configuration files (continued)

Directory under *BASEDIR*	Filename	User editable	Description
smarts/conf/icoi	profileconfig-sample.xml	Yes	Sample XML file for user profile configuration. It can be used as a template for creating additional files.
smarts/conf/icoi	userconfig-sample.xml	Yes	Sample XML file for user configuration. It can be used as a template for creating additional files.

Service Assurance Adapter Platform architecture

The Adapter Platform consists of four basic components:

- Service Assurance Adapter Platform Server

  The Adapter Platform Server prepares event information for use by the Global Manager. It normalizes the event information and places it in a context understood by the Global Manager.

- Syslog Adapter

  The Syslog Adapter parses a system log file and generates notifications based on the contents of the file. Either the complete contents of the file are read or the file can be tailed, which means that only newly added information is read.

- SNMP Trap Adapter (Receiver)

  The SNMP Trap Adapter parses SNMP traps received and generates notifications based on the contents of the traps. The SNMP Trap Adapter can be configured to handle many different types of traps.

  If you plan to receive SNMPv3 traps, additional steps are required. Configuring the SNMP Trap Adapter to receive SNMPv3 traps provides more information.

- Command line interface sm_ems)

  The command line interface creates or modifies notifications that are sent to the Global Manager. This interface may be used in conjunction with third-party applications to send events to the Global Manager.

  Note The Syslog Adapter and the SNMP Trap Adapter configurations must be customized to specify the relevant file contents or the SNMP traps to use and how the information maps to objects in the Global Manager repository.

Topology Importer

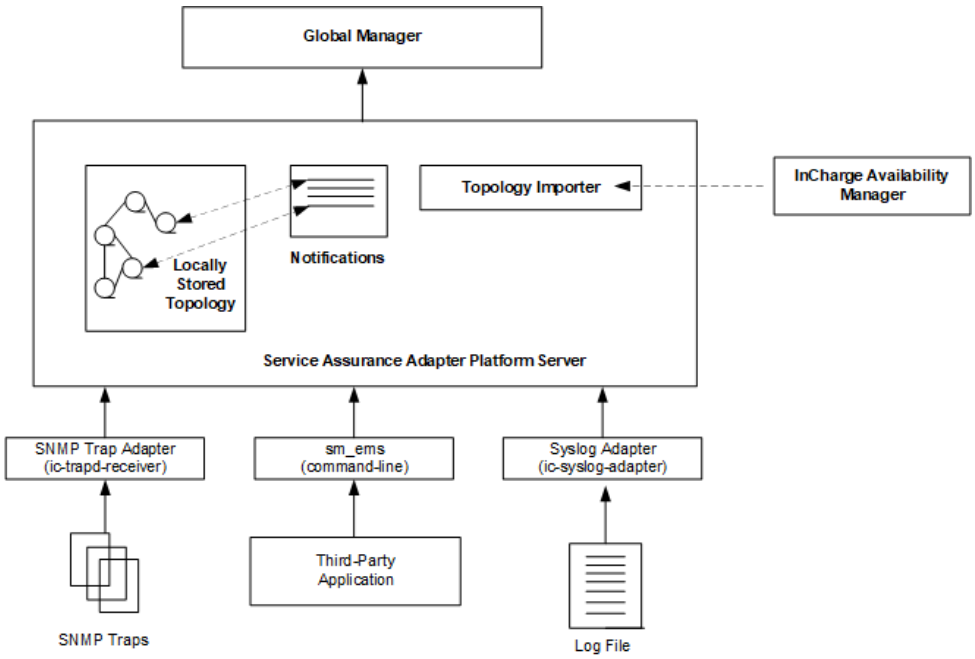
In addition to the basic Adapter Platform components, you can enable an Adapter Platform feature, the Topology Importer, which provides a consistent method of identifying the systems associated with Adapter Platform events.

The Adapter Platform creates notifications from different sources and tries to associate these notifications to the topology element where the notification occurred. Making this association can be difficult because there are many different methods of identifying topology elements.

The Topology Importer collects IP addresses and hostnames from an IP Availability Manager so the Adapter Platform can accurately place events in their topological context.

The Adapter Platform updates this list as the information changes in the IP Availability Managers. Service AssuranceAdapter Platform architecture shows the Service Assurance Adapter Platform architecture.

Figure 2-1. Service Assurance Adapter Platform architecture



Overview of configuration for Adapter Platform

Configuring the Adapter Platform to import and normalize event and topology information requires that you complete one or more of the following procedures. Not all of these configuration procedures are relevant in all Adapter Platform deployments.

Depending on which components you are implementing, you can omit some of these steps:

- Configure the Adapter Platform Server
- Configure the SNMP Trap Adapter
- Configure the Syslog Adapter
- Configure the Adapter Platform Server and adapters to start and stop automatically.

The following configuration sections describe each of these procedures.

Configuring Adapter Platform Server

Configuring the Adapter Platform Server involves:

- Configuring notification lists
- Configuring user profiles
- Configuring users
- Configuring system defaults
- *System defaults cannot be deleted. on page 32*
- Configuring domain types

Configuring notification lists

Topics in this section include:

- Creating a notification list
- Modifying a notification list
- Disabling a notification list
- Configuring custom notification lists

Creating a notification list

To create a notification list by using the Notification List Creation Wizard:

1 From the **Global Manager Administration Console**, select **Edit** > **New Notification List**.

The **Notification List Creation Wizard** appears.

2 Type a unique name for the **Notification List**.

3 Select to create a new notification list or copy an existing notification list:

- If a new notification list is being created, all of the filter properties are empty.
  - If you copy a notification list, the notification list properties contain the same values as the copied list.

4 Click **Next**.

5 Create an expression filter or type the name of an ASL Filter prefaced by the name of the *BASEDIR*/smarts/local/rules directory in which it is located, and click **Next**.

6 Edit the column headings that are displayed in the notification log. The left column lists the attributes included in a notification. The right column lists the column names as they are currently displayed. You can edit the values in the right column by double-clicking a field.

7 Perform one of the following:

- Click **Next** to view the confirmation panel.
  - Click **Finish** to create the notification list.

    The new notification list is displayed in the Global Manager Administration Console.

Modifying a notification list

To edit the filters and the column headings for a notification list:

1 Select a notification list in the tree.

The **Configure Notification List** panel appears.

2 Click **Edit Filter** to modify the filter.

The **Edit Filter** dialog box appears. The Service Assurance Manager Configuration Guide provides additional information.

3 Click **OK** when finished editing the filters.

4 Edit the display headings as needed. Double-click the display heading in the right column to edit the text. The new display heading is visible when users reattach to the Global Manager.

5 Click **Apply**.

Note To provide a value for a UserDefined attribute, you must configure a hook script to populate the field. *Service Assurance Manager Configuration Guide* provides information about hook scripts.

Disabling a notification list

You can disable a notification list if you do not want it active, but do not want to delete it. When a user attaches to a Global Manager and their notification is disabled, their console will not receive any notifications. The console will receive notifications when the notification list is enabled.

To disable a notification list:

1 Select the list from the tree in the Global Manager Administration Console.

2 Clear the **Enabled** checkbox.

3 Click **Apply**.

The name of the disabled notification list appears dimmed.

Configuring custom notification lists

You do not need to create a notification list in the Adapter Platform Server for the Global Manager to use. By default, it connects to the Adapter Platform Server's Default notification list. However, if you want to use a custom notification list, you can create it through the Global Manager Administration Console, or you can create it through the use of the nlconfig-sample.xml file and import it with the **sm_config** command. This allows you to send a subset of notifications from the Adapter Platform Server to the Global Manager. Then you need to modify the local copy of the dxa-oi.conf on the Global Manager and change the following line:

```
sub    Default/n
to
sub     <notification_list>/n
```

Configuring user profiles

Topics in this section include:

- Creating a user profile
- Deleting a user profile
- Modifying a user profile

Creating a user profile

When using the User Profile Creation Wizard, you are prompted to specify the users, notification list, tools, saved consoles, and console operations associated with the user profile.

Note You can create users and notification lists from within the User Profile Creation Wizard.

As you proceed through the wizard, you may notice that certain elements, such as users or tools, appear dimmed. This indicates that these elements are not enabled. You can associate disabled elements with a user profile. However, such elements are not available to the user profile. For example, if a tool is not enabled, it will not be available to a user until it is enabled. You can enable or disable elements by selecting them in the tree, and selecting or clearing the Enable box in the Configure User Profile panel.

In addition, you can click Finish at any point and the wizard will create the user profile. If you are creating a new user profile, any options after the point where you selected Finish will be blank. If you copied an existing user profile, the new profile will include the options of the copied profile.

Note Users may have to detach and reattach for changes to their user profile to take effect.

To create a user profile:

1 From the Global Manager Administration Console, click the **Launch User Profile Wizard** toolbar button.

The **User Profile Creation Wizard** appears.

2 Type the name of the user profile and select whether to create a new profile or to copy an existing profile, and click **Next**. If you want to copy an existing profile, the remaining configuration options will be identical to the user profile that was copied.

The name of a user profile cannot be the same as the name of a user. For example, you cannot create a user profile Operator and a user Operator. A suggested naming convention is to name the user profile with a –profile extension. For example, you could have a user profile called Operator–profile.

3 Select a notification list from list box and click **Next**. Optionally, you can select New Notification List from the menu to create a new notification list.

4 Select the user or users associated with this user profile. You can select multiple users by holding the **Ctrl** key when selecting users. Click **Add** to associate the user or users with the user profile.

If the user does not exist, you can create a user by doing the following:

5 Type the username in the **Create and add new user** field.

6 Click **Add**.

7 Click **Next**.

Note A user can belong to only one user profile. If the user you select is already a member of another user profile, the user is associated with the new user profile.

8 Select any appropriate server tools from the list of Available server tools and click **Add** to associate them with the user profile. You can select multiple tools by holding the **Ctrl** key while selecting. When finished, click **Next**.

Note Click the **Allow user to launch client or server tools only for a single selected object** option to allow users to only launch client or server tools for a single selected object.

9 Select one or more appropriate client tools from the list of Available client tools, and click **Add** to associate them with the user profile. You can select multiple tools by holding the **Ctrl** key while selecting. When finished, click **Next**.

Note Click the **Allow user to launch client or server tools only for a single selected object** option to allow users to only launch client or server tools for a single selected object.

10 Select the saved consoles that are opened when users attach to the Global Manager, and click **Next**. If a saved console is not selected, the default NotificationLog console will appear when the user attaches to the Global Manager.

If the appropriate console is not listed, type the name of the console in the **Add new console** field, and click **Add**. You can specify the name of a saved console whether or not the saved console exists.

11 Select the console operations for users associated with this user profile. You can select **Other** for specific console operations or select one of the default sets of console operations:

- Read Only
- Operator
- Administrator

The default sets of console operations cannot be modified.

You can assign a modified version of one of the default sets of console operations by first selecting the set that provides most of the console operations to assign. Then, select **Other**. When you click **Next**, the console operations that correspond to the set you chose are selected but you can enable or disable individual console operations.

12 Click **Next**. The final screen of the **User Profile Wizard** shows all newly created elements, including the new user profile. By default, the user profile is enabled. You can disable it by clearing the box next to the profile name.

13 Click **Finish** to display a dialog box confirming that the user profile elements were created.

Deleting a user profile

To delete a user profile:

1 Select the user profile.

2 Select **Edit** > **Delete**.

You can disable a user profile without removing it. Disabling a user profile describes how to disable a user profile.

Note You must not delete the default-profile user profile. When users attach to a Global Manager and their username is not assigned to a user profile, the default-profile user profile is applied. If the default-profile does not exist, an error message appears and the Global Console will not open.

Modifying a user profile

To modify a user profile:

1 Select the user profile.

2 Modify the properties of the user profile in the right panel:

3 Change the notification list.

4 Add or remove users.

5 Add or remove server tools.

6 Add or remove client tools.

7 Add or remove saved consoles.

8 Add or remove console operations.

9 Click **Apply**.

Changes to a user profile may not be available to console users until they restart the Global Console.

Disabling a user profile

You can disable a user profile without having to delete it. When users of a disabled profile attach to the Global Manager, they are assigned the default-profile. If the default-profile is deleted, users associated with a disabled user profile cannot attach to the Global Manager.

To disable a user profile:

1 Select the user profile.

2 Clear the **Enabled** checkbox.

3 Click **Apply**.

Changing the notification list for a user profile

A notification list determines what notifications a user sees in the Global Console. Changing the notification list associated with a user profile affects all the users of this profile.

To change the notification list for a user profile, select a new notification list from the Notification List menu.

Adding or removing a user

A user can be associated with only one user profile. If you add a user to a user profile, the user is automatically removed from its previous user profile.

To add a user to a profile:

1 Select the user profile from the tree.

2 From the **Configure User Profile** panel, click **Modify List** in the **Users** section.

This displays the **Modify Users** dialog box.

3 Add or remove users from the user profile.

4 Click **OK**.

5 Click **Apply**. Creating a user profile describes how to create a user account.

To remove a user, select the user profile in the tree. From the Configure User Profile panel, select the user from the Users list and click Remove Selected. Click Apply at the bottom of the Configure User Profile panel.

When you remove a user from a user profile, the user is automatically associated with the default-profile until you assign the user to a different user profile.

Adding or removing a tool

Client and server tools are configured separately, however the process for adding or removing a client or server tool from a user profile is the same.

To add a tool to a user profile:

1 Select the user profile in the tree.

2 From the **Configure User Profile** panel, click **Modify List** in the **Server Tools** or **Client Tools** section.

The **Modify Server Tools** or **Modify Client Tools** dialog box appears.

3 Add or remove tools from the user profile.

4 Click **OK**.

5 Click **Apply.**

To remove a tool:

6 Select the user profile in the tree.

7 From the **Configure User Profile** panel, select a tool and click **Remove Selected**.

8 Click **Apply** at the bottom of the **Configure User Profile** panel.

Adding or removing a saved console

A saved console provides users with a preconfigured view of the network, helping users focus on aspects of the managed system they are to monitor.

To add a saved console to a user profile:

1 Select the user profile in the tree.

2 From the **Configure User Profile** panel, click **Modify List** in the **Saved Consoles** section.

The **Modify Saved Consoles** dialog box appears.

3 Add or remove saved consoles from the user profile.

4 Click **OK**.

5 Click **Apply**.

To remove a saved console:

6 Select the user profile in the tree.

7 From the Configure User Profile panel, select a saved console and click **Remove Selected**.

8 Click **Apply** at the bottom of the **Configure User Profile** panel.

Adding or removing console operations

Console operations are the functions and commands a user can invoke through the Global Console. By adding or removing console operations, an administrator can determine the level of access a user has to console operations.

To add or remove console operations for a user profile:

1 Select the user profile in the tree.

2 From the **Configure User Profile** panel, click **Modify List** in the Console Operations section.

The **Console Operations** dialog box appears.

3 Select or clear console operations for the user profile.

Note In addition, you can select one of the three default groups of console operations and apply it to the user profile.

4 Click **OK** to close the dialog box.

5 Click **Apply** at the bottom of the **Configure User Profile** panel.

Configuring users

Topics in this section include:

- Creating a user account
- Deleting a user account

Creating a user account

You can create new users in one of the following ways:

- Through the User Profile Wizard when you create a user profile as described in Creating a user profile.
- Through the User Wizard as described below.
- Through the **sm_config** utility as described in The sm_edit utility.

  To create a new user with the User Wizard:

  a Select the **Launch User Wizard** toolbar button.

  The **User Creation Wizard** appears.

  b Type the name of the new user. The username must be unique. Click **Next**.

  c Select a user profile for this user from the list box.

  d Do one of the following:

  - Click **Next** to display a confirmation screen.
  - Click **Finish** to create the user.

e Add the username and password in the serverConnect.conf file used by the Global Manager.

Deleting a user account

To delete a user:

1 Select the user in the tree.

2 Select **Edit** > **Delete**.

Note Do not disable or remove the Admin user account. If the Admin user account is removed or disabled, but other users still have the same privileges, none of the applications can be shut down.

Configuring system defaults

To configure system defaults:

1 Open the **Global Manager Administration Console**.

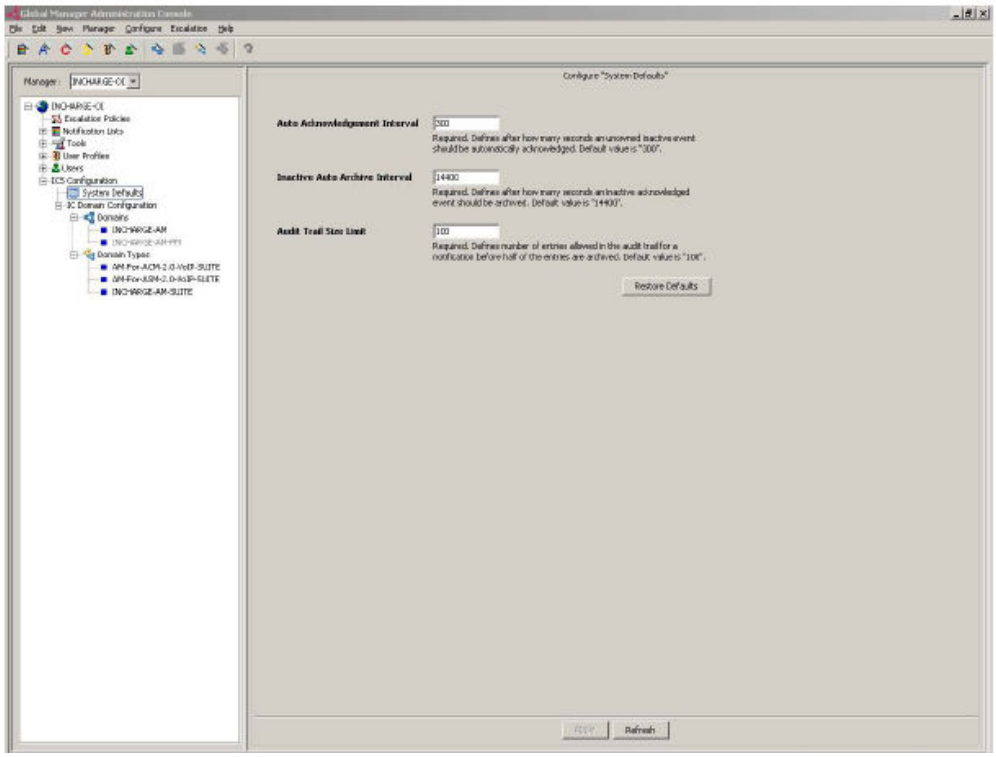
2 From the **Manager** list, select the Adapter Platform (INCHARGE-OI).

3 Expand **ICS Configuration.**

4 **Click System Defaults**.

The **Configure System Defaults** interface appears, as illustrated in Global Manager Administration Console — Configure System Defaults.

Figure 2-2. Global Manager Administration Console — Configure System Defaults



5 Type or select your system defaults configuration options.

6 Click **Apply**.

System default parameters describes the system default parameters.

Table 2-2. System default parameters

Parameter	Description
Auto acknowledgement interval	Defines interval, in seconds, after which an inactive and unowned notification is acknowledged. Notifications that are acknowledged by the Global Manager are owned by the user SYSTEM. **Default**: 300 seconds
Inactive auto archive interval	Defines interval, in seconds, after which an inactive and acknowledged notification is archived. Default is 14400 seconds (4 hours). If this value is set to zero, archiving is disabled and notifications will not be deleted, causing Global Manager to use more memory.
Audit trail size limit	Defines the number of audit log entries for each notification that are saved and visible in the Global Console before the log contents are archived. When this limit is reached, half of the entries are written to the notification archive. **Default**: 100 entries

Note System defaults cannot be deleted.

Configuring domains

Topics in this section include:

- Creating a domain
- Modifying domains
- Deleting domains

Creating a domain

To create a domain:

1 From the **Global Manager Administration Console**, select **Edit** > **New Domain**.

The **Domain Creation Wizard** appears.

2 Type the name of the domain server in the **Domain Name** field.

3 Type the description of the domain in the **Domain Description** field, and click **Next**.

Note To create a domain by using the **Copy Existing** option, select an existing domain server from the **Copy Existing** list, type a **Domain Name**, and click **Finish**.

4 Select the domain type. Available options include:

- **Use Default Type**

  Note This option is available only when copying an existing domain.

- **Create New Type**

  If this option is selected, type the name of the domain type in the **Type Name** field.

  To copy an existing domain type, select an existing domain type from the **Copy Existing** list.

- **Select Existing Type**

  If this option is selected, select a domain type from the **Select Type** list.

5 Click **Next**.

6 Edit the domain settings, and click **Next**.

Domain settings defines the available domain settings.

Table 2-3. Domain settings

Option	Definition
DXA file	Defines the DXA configuration file.
Hook script: enable	Enables the hook script. **Note** If the hook script is enabled, type the path for the hook script in the field.
Minimum certainty	Defines the minimum certainty. **Default value**: 0.01
Smoothing interval	Defines the smoothing interval. **Default value**: 65

The instance that is created when the domain creation process is complete appears.

Note By default, the domain is enabled. If the domain is part of a domain group, it is only enabled if the domain group it belongs to is enabled. To disable the domain, clear its checkbox.

1 Click **Finish**.

Modifying domains

To modify a domain:

1 Open the **Global Manager Administration Console**.

2 From the **Manager**List, select the Adapter Manager (INCHARGE-OI).

3 Expand **ICS Configuration**.

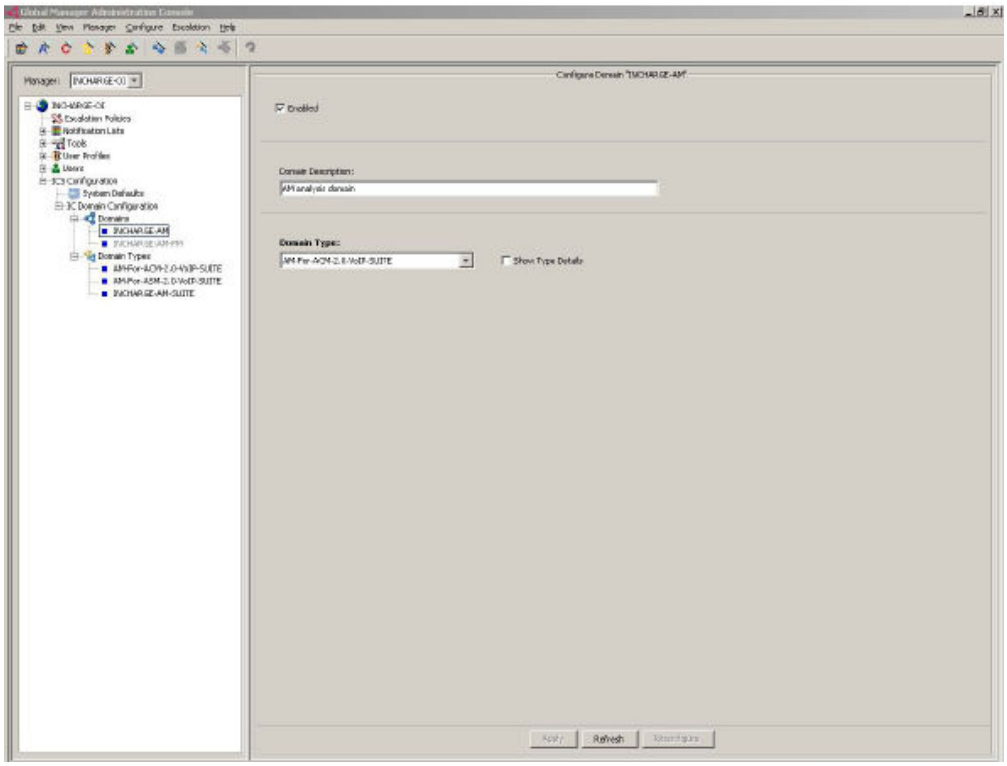
4 Expand **IC Domain Configuration**.

5 Expand **Domains**.

6 Click the appropriate domain.

The **Configure Domain**interface appears, as illustrated in

Figure 2-3. Global ManagerAdministration Console — Configure Domain interface



7 Type the appropriate configurations.

Domain parametersdescribes the available domain parameters.

Global ManagerAdministration Console — Configure Domain interface.

Table 2-4. Domain parameters

Parameter	Description
Enable	Enables the domain.
Domain description	Provides a description of the domain.
Domain type	Defines the domain type.
Show type details	Displays the domain type information.

1 Click **Apply**.

2 Click **Reconfigure**.

Note Configuration changes are not applied until **Reconfigure**is clicked. Clicking **Reconfigure**starts the data synchronization process. Click **Reconfigure**only after all updates to domains, domain tags, domain groups, and domain types have been made.

Deleting domains

To delete domains:

1 Open the **Global Manager Administration Console**.

2 From the **Manager** list, select the Adapter Platform (INCHARGE-OI).

3 Expand **ICS Configuration**.

4 Expand **IC Domain Configuration**.

5 Expand **Domain**.

6 Click the appropriate domain.

7 Select **Edit** > **Delete**.

The selected domain is deleted.

Note Domains that refer to other domains cannot be deleted.

Configuring domain types

Topics in this section include:

- Creating domain types
- Modifying domain types
- Deleting domain types

Creating domain types

To create a domain type by using the Domain Groups Creation Wizard:

1 From the **Global Manager Administration Console**, select **Edit** > **New Domain Type**.

The **Domain Groups Creation Wizard** appears.

2 Select a type to be copied from the **Copy Existing** list.

3 Type a name in the **Type Name** field.

4 Type a description in the **Type Description** field.

5 Click **Next**.

6 Edit the domain type settings, and click **Next**.

Domain type settings defines the available domain type settings.

Table 2-5. Domain type settings

Option	Definition
DXA file	Defines the DXA configuration file.
Hook script: enable	Enables the hook script. **Note** If the hook script is enabled, type the path for the hook script in the field.
Minimum certainty	Defines the minimum certainty. **Default value**: 0.01
Smoothing interval	Defines the smoothing interval. **Default value**: 65

1 Select the servers from the **Available Domains** list.

2 Click **Add** to move the server to the **Selected Domains** list.

This defines the servers associated with the new domain type.

Note Domains cannot be removed from a domain type — they can only be reassigned to another domain type. By adding domains, you are only reassigning the domains to a different domain type. Newly added domains appear in green text. You can only delete domains displayed in green text. After clicking **Apply**, the domains cannot be deleted.

3 Click **Next**.

Objects that will be created after the domain type creation process is complete.

4 Click **Finish**. The configuration objects are created in the server.

Modifying domain types

To modify a domain type:

1 Open the **Global Manager Administration Console**.

2 From the **Manager** list, select the Adapter Platform (INCHARGE-OI).

3 Expand **ICS Configuration**.

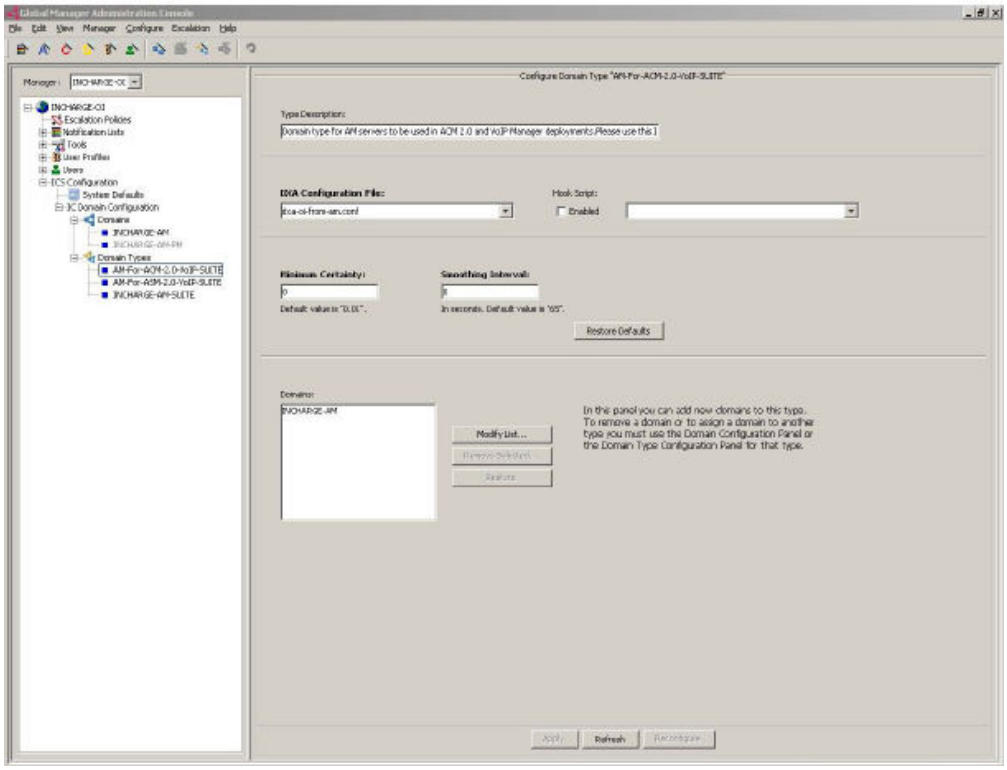
4 Expand **IC Domain Configuration**.

5 Expand **Domain Types**.

6 Click the appropriate **INCHARGE Domain Type**.

The **Configure Domain Type** interface appears, as illustrated in Global Manager Administration Console — Configure Domain Type interface.

Figure 2-4. Global Manager Administration Console — Configure Domain Type interface



7 Type the appropriate configurations.

Domain type configuration parameters defines the available domain type configuration parameters.

Table 2-6. Domain type configuration parameters

Parameters	Definition
Type description	Defines the domain type.
DXA configuration file	Defines the DXA configuration file.
Hook script: enable	Enables the hook script. **Note** If the hook script is enabled, type the path for the hook script in the field.
Minimum certainty	Defines the minimum certainty. **Default value**: 0.01

Table 2-6. Domain type configuration parameters (continued)

Parameters	Definition
Smoothing interval	Defines the smoothing interval. **Default value**: 65
Domains	Defines the domains associated with the domain type. **Note** Click **Modify List** to add domains, select a domain from the list and click **Remove Selected** to remove domains, and click **Restore** to restore previously configured domains. Domains cannot be removed from a domain type, they can only be reassigned to another domain type.

1 Click **Apply**.

2 Click **Reconfigure**.

Note Configuration changes are not applied until **Reconfigure** is clicked. Clicking **Reconfigure** starts the data synchronization process. Click **Reconfigure** only after all updates to domains, domain tags, domain groups, and domain types have been made.

Deleting domain types

To delete domain tags:

1 Open the **Global Manager Administration Console**.

2 From the **Manager** list, select the Adapter Platform (INCHARGE-OI).

3 Expand **ICS Configuration**.

4 Expand **IC Domain Configuration**.

5 Expand **Domain Types**.

6 Click the appropriate domain type.

7 Select **Edit** > **Delete**.

The selected domain type is deleted.

Note Domain types that refer to other domains cannot be deleted.

Configuring security for the Adapter Platform

It is important to secure access to the Adapter PlatformServer. Service Assurancecomponents authenticate users and determine their privileges through three files: serverConnect.conf, clientConnect.conf, and brokerConnect.conf. These files ensure that only authorized users access Service AssuranceServer applications. Use **sm_edit**to modify these files, which are located in *BASEDIR*/smarts/conf.

The Adapter Platformfunctions as a server application. The following are possible clients of the Adapter PlatformServer:

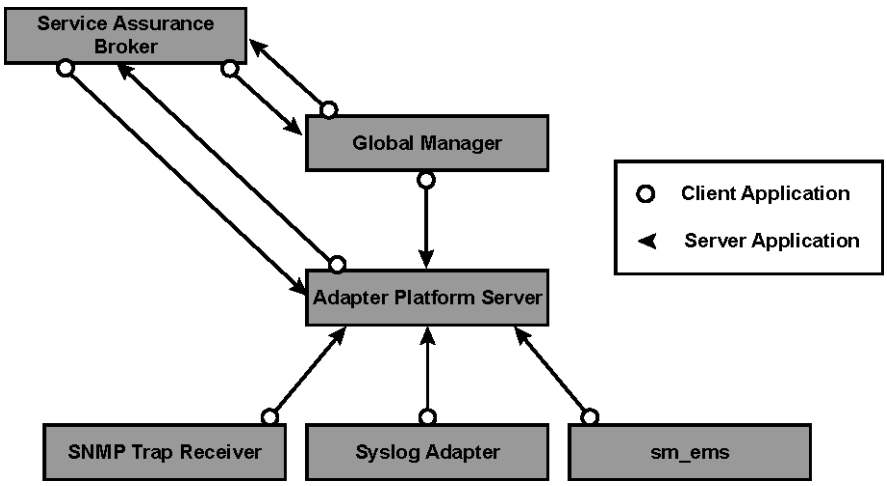
- Broker
- Command line utilities (dmctl)
- Global Manager
- sm_ems
- SNMP Trap Adapter(Receiver)
- Syslog Adapter

System Administration Guideprovides a complete list of server and client programs and a more detailed explanation of the Service Assurancesecurity mechanism.

Security configuration of the Adapter Platformprovides an overview of the security files required in a typical deployment of the Adapter Platform. A circle in the figure indicates that the component represented by the adjacent box is a client application. An arrowhead indicates that the component represented by the adjacent box is a server application. For example, the Adapter Platformis a client of the broker and the Adapter Platformhas five clients"the broker, the Global Manager, the SNMP Trap Adapter, the Syslog Adapter, and sm_ems. Although not depicted in the figure, the SNMP Trap Adapter, the Syslog Adapter, and sm_ems are also clients of the Broker because everything is a client of the Broker.

Note In general, the Global Managerand the Adapter Platformare deployed on different hosts. If the Adapter Platformresides on the same host as another Service Assuranceserver application, such as the Adapter Platform, then they can share the same serverConnect.conf file. Likewise, if clients of the Adapter Platformreside on the same host as other Service Assuranceclient applications, they too can share the same clientConnect.conf file.

Figure 2-5. Security configuration of the Adapter Platform



The following sections provide examples of authentication records for the client programs, SNMP Trap Adapter(Receiver), Syslog Adapter, and **sm_ems**command line interface and corresponding records for the server program, Adapter PlatformServer.

The System Administration Guideprovides examples of authentication records for the broker and the Global Manager.

SNMP Trap Adapter security

The SNMP Trap Adapter is a client of the Adapter Platform Server. As such it will use information in the clientConnect.conf file to provide authentication information to the Adapter Platform Server. The Adapter Platform Server will compare this information to entries in the serverConnect.conf file to validate the connection. Thus, both these files must be configured.

The clientConnect.conf file must be configured on the host where the SNMP Trap Adapter runs. In the clientConnect.conf file you should create an entry for the system user ID running the SNMP Trap Adapter with a corresponding username and password. For example:

```
trap-uid1 : INCHARGE-OI : admin : changeme
```

where:

- trap-uid1 is the system user ID running the SNMP Trap Adapter
- INCHARGE-OI is the Adapter Platform Server name
- admin is the login username
- changeme is the login password

The serverConnect.conf must be configured on the same host as the Adapter Platform. In the serverConnect.conf, an entry must exist that corresponds to the username and password specified in the clientConnect.conf file for the system user ID running the SNMP Trap Adapter. The user must be given All privileges. For example:

```
INCHARGE-OI : admin : changeme : ALL
```

where:

- INCHARGE-OI is the Adapter Platform Server name (you could use an * to indicate all Service Assurance servers)
- admin is the login username
- changeme is the login password
- ALL is the privileges granted this user

The SNMP Trap Adapter does not have to run on the same host as the Adapter Platform. System Administration Guide provides more information about security configuration.

Syslog Adapter security

The Syslog Adapter is a client of the Adapter Platform Server. As such it will use information in the clientConnect.conf file to provide authentication information to the Adapter Platform Server. The Adapter Platform Server will compare this information to entries in the serverConnect.conf file to validate the connection. Thus, both these files must be configured.

The clientConnect.conf file must be configured on the host where the Syslog Adapter runs. In the clientConnect.conf file you should create an entry for the system user ID running the Syslog Adapter with a corresponding username and password. For example:

```
syslog-uid1 : INCHARGE-OI : admin : changeme
```

where:

- syslog-uid1 is the system user ID running the Syslog Adapter
- INCHARGE-OI is the Adapter Platform Server name
- admin is the login username
- changeme is the login password

Note The system user ID that starts the Syslog Adapter will need administrative privileges.

The serverConnect.conf must be configured on the same host as the Adapter Platform. In the serverConnect.conf, an entry must exist that corresponds to the username and password specified in the clientConnect.conf file for the system user ID running the Syslog Adapter.

The user must be given All privileges. For example:

```
INCHARGE-OI : admin : changeme : ALL
```

where:

- INCHARGE-OI is the Adapter Platform Server name (you could use an * to indicate all Service Assurance servers)
- admin is the login username
- changeme is the login password
- ALL is the privileges granted this user

The Syslog Adapter runs on the same host as the Adapter Platform. The System Administration Guide provides more information about security configuration.

sm_ems command line interface security

The **sm_ems** command line interface may or may not run on a host different from the Adapter Platform Server. The clientConnect.conf file must be configured on the host running the command line interface. You should create a unique username and password that matches a user in serverConnect.conf and give that user All privileges. Typically, when using the command line interface from a third-party application, the username and password should not be prompted. However, if you want to run **sm_ems** manually, then prompting could be used. The System Administration Guide provides more information about security configuration.

Starting and stopping the Adapter Platform Server

If you installed the Adapter Platform Server as a service, it automatically starts when the system starts up. The following instructions describe how to use the **sm_service** utility to manually start and stop the Adapter Platform Server.

Note To use the **sm_service** utility to install a service or start a service, you must have administrative privileges on the local host.

Issue one of the following from the command line:

```
BASEDIR/smarts/bin/sm_service start ic-icoi-server
```

or

```
BASEDIR/smarts/bin/sm_service stop ic-icoi-server
```

The System Administration Guide provides more information about the **sm_service** utility or about how to modify **sm_service** options.

Default sm_service parameters for Adapter Platform Server

When the Adapter Platform Server is installed as a service during the installation process, the Adapter Platform Server is set up with default parameters. The default parameters and their values are:

```
t /InCharge/SAM/smarts/bin/sm_service install
--startmode=runonce --name=ic-icoi-server
--description="SMARTS SAM Adapter Platform Server"
/C:/InCharge/SAM/smarts/bin/sm_server
--name=INCHARGE-OI
--config=icoi
--port=0
--ignore-restore-errors
--output s
```

The sm_edit utility

As part of the VMware Smart Assurance deployment and configuration process, you need to modify certain files. User modifiable files include configuration files, rule set files, templates, and files (such as seed files and security configuration files) that contain encrypted passwords. Original versions of these files are installed into appropriate subdirectories under the *BASEDIR*/smarts/ directory. Examples follow:

Examples

- The original versions of the Global Manager configuration files on UNIX are installed to /opt/InCharge/SAM/smarts/conf/ics
- Original versions of files should not be altered. If a file requires modification, it must be stored as a local copy of the file in *BASEDIR*/smarts/local or one of its subdirectories. For example:

  Examples

- A modified dxa-sam.conf file on UNIX should be saved to /opt/InCharge/SAM/smarts/local/conf/ics
- VMware Smart Assurance software is designed to first search for user modifiable files in *BASEDIR*/smarts/local or one of its subdirectories. If a modified version of a file is not found in the local area, the software then searches appropriate nonlocal directories.

  To facilitate proper file editing, VMware, Inc. provides the **sm_edit** utility with every product. When used to modify an original version of a file, this utility automatically creates a local copy of the file and places it in the appropriate location under *BASEDIR*/smarts/local. This ensures that the original version of the file remains unchanged. You can invoke **sm_edit** from the command line.

  To invoke the *s* **m_edit** utility from the command line, specify the path and the name of the file to edit under *BASEDIR*/smarts. If multiple products are running on the same host, invoke **sm_edit** from the bin directory of the product with the files to edit.

  Example

  To edit the configuration file for the Global Manager, invoke the **sm_edit** utility as follows:

  ```
  # /opt/InCharge/SAM/smarts/bin>sm_edit conf/ics/dxa-sam.conf
  ```

  In this example, the **sm_edit** utility automatically creates a local copy of the dxa-sam.conf file in the /opt/InCharge/SAM/smarts/local/conf/ics directory and opens the file in a text editor. If a local version of the file already exists, the **sm_edit** utility opens the local version in a text editor. In addition, **sm_edit** creates any necessary subdirectories.

  The System Administration Guide provides additional information about the **sm_edit** utility.

VMware Smart AssuranceSNMP Trap Adapter

3

Read the following topics next:

Overview

The Simple Network Management Protocol (SNMP) Trap Adapter (Receiver) collects and parses SNMP traps and generates notifications to the Global Manager based on the contents of the traps. The Trap Adapter can receive SNMP v1, v2c, or v3 traps. However, it forwards only v1 or v2c traps. If SNMPv3 traps are received, they are converted to one of the other formats before being sent to a Domain Manager or to another trap receiver.

Two deployment scenarios are possible. The Service Assurance Manager Deployment Guide provides more information.

Scenario 1: Single trap receiver associated with Adapter Platform

In Scenario 1, a single trap receiver listens for all traps from the network.

Note Do not use this scenario in a production environment! This scenario may be employed for test or debug purposes when forwarding small numbers of traps to the Adapter Platform and possibly one other manager such as IP Availability Manager.

In this scenario, you must edit the following files:

- The InCharge/SAM/smarts/conf/icoi/trapd.conf file to indicate the port and hostnames to forward the traps to. Configuring trapd.conf (trap exploder and trap receiver) provides more information.

- If you plan to receive SNMPv3 traps from the network, you must edit a seed file as explained in Editing seed files load SNMPv3 agent credentials.

  Note This trap receiver (ic-trapd-receiver) is the one that starts automatically if you install the SNMP Trap Adapter as a service. Default sm_service parameters for the SNMP Trap Adapter provides more information.

- The InCharge/SAM/smarts/conf/icoi/trap-mgr.conf file to convert traps to notifications and send them on to the Adapter Platform. Configuring the SNMP Trap Adapter to forward notifications provides more information.
- The ics-default.xml file to allow communication (forwarding of notifications) between the Adapter Platform and the Global Manager.

Scenario 2: Trap exploder forwards traps to a second trap receiver

In Scenario 2, there are two trap receivers. The first trap receiver, referred to as the trap exploder, receives all traps from the network and selectively forwards only those traps that meet certain configuration requirements. This scenario is the recommended deployment scenario. It may be employed when a large volume of traps comes from the network and you need to send traps to multiple Domain Managers as well as to a second trap receiver associated with Adapter Platform.

To set up the trap exploder, configure the following files:

- Edit *BASEDIR*/smarts/conf/trapd/trapd.conf and specify the port to forward traps to as well as the hostname of the Domain Managers and Adapter Platform. (*BASEDIR* may be either InCharge/IP/ or InCharge/SAM/. The same trapd.conf file is shipped with both products.) Configuring trapd.conf (trap exploder and trap receiver) provides more information.
- If you plan to receive SNMPv3 traps from the network, you must edit a seed file as explained in Editing seed files in order to load SNMPv3 agent credentials.
- Copy this seed file to the *BASEDIR*/smarts/conf/trapd path in the product (IP or SAM) where the trap exploder is running. For example, to start the trap exploder from the Service Assurance Manager:

  ```
  tIncharge/SAM/smarts/bin>sm_trapd --port=162 --rules=default --name=TRAP_EXPLODER --
  seed=seedfiles
  ```

  With SAM 9.2, the behaviour of sm_trapd option, --rules=default is changed so that it will no longer log received traps by default.

  To enable logging of received traps, follow the procedure:

  a Goto BASEDIR/smarts/rules/trapd and type the following command to edit the *trapParse.asl* file:

  ```
  ./sm_edit <BASEDIR>/smarts/rules/trapd/trapParse.asl
  ```

b Type the following command to change the setting of debug from FALSE to TRUE on line 25 in the trapParse.asl file:

```
debug = TRUE; /* Set to TRUE to enable printing of trap data */
```

c Save the modified file and then restart sm_trapd.

To start the trap exploder as a service you must carry out a manual step in which you need to ensure the name is unique.

For example:

Note t s indicates the command must be typed as one line. Type the command from the *BASEDIR*/smarts/bin directory for the Service Assurance Manager product.

```
t /InCharge/SAM/smarts/bin/sm_service install --force --unmanaged
--startmode=runonce
--description="VMware Smart Assurace SNMP Trap Exploder Server"
--name=trap_exp
 /InCharge/SAM/smarts/bin/sm_trapd
    --name=TRAP_EXPLODER
    --config=trapd
    --port=162
    --sport=9180
    --seed=seedfile
    --rules=default
    --output s
```

To configure the second trap receiver (associated with Adapter Platform), edit both files listed below:

- InCharge/SAM/smarts/conf/icoi/trapd.conf, as described in Configuring trapd.conf (trap exploder and trap receiver).
- InCharge/SAM/smarts/conf/icoi/trap-mgr.conf file, as described in Configuring the SNMP Trap Adapter to forward notifications.
- Start the trap receiver associated with the Adapter Platform as a service (if not already installed as a service). Default sm_service parameters for the SNMP Trap Adapter provides more information.

Configuring the SNMP Trap Adapter to receive SNMPv3 traps

You may want to receive SNMPv3 traps from some of your network devices. The SNMP Trap Adapter supports the following SNMPv3 features:

- The ability to receive SNMPv3 traps, convert them to SNMPv2c, and forward them in SNMPv2c format to other elements of the system such as IP Domain Managers or Adapter Platform.

- The use of the SNMPv3 User Security Model (USM) for authentication and privacy, as described in RFC-3414.
- The ability to load USM user credentials by using text files (similar in format to existing IP domain manager seed files).
- The use of authentication protocols MD5 and SHA-1 (RFC-3414).
- The use of privacy protocols DES (RFC-3414) and AES-128 (RFC 3826).

  If the SNMPv3 network devices are configured to support authentication and encryption, you have to load the agent credentials by using a seed file to enable the SNMP Trap Adapter to receive the SNMPv3 traps.

Configuring the seed file to load SNMPv3 credentials

A seed file is used to load the SNMPv3 User Security Model (USM) data into the Local Credentials Database (LCD). The SNMP Trap Adapter recognizes SNMPv3 device entries by the engineID/ userName pair for a given device listed in the seed file.

A sample seed file is shipped with the Service Assurance Manager software in the InCharge/SAM/ smarts/conf/icoi path. The format (syntax) of the seed file used for the SNMP Trap Adapter is the same as it is for an IP domain manager; however, there may be slight variations in field values when used for IP discovery. The IP Manager User Guide provides specific seed file values required for the discovery process.

Editing seed files provides more details about editing seed file entries for use by the SNMP Trap Adapter.

If you plan to use the seed file for both IP discovery and for the SNMP Trap Adapter Scenario 1 (single trap receiver only), then ensure the seed file resides in two places:

- InCharge/IP/smarts/conf
- InCharge/SAM/smarts/conf/icoi

  If you plan to use the seed file for the SNMP Trap Adapter Scenario 2 (trap exploder), then ensure the seed file is in the following path (not in the InCharge/SAM/smarts/conf/icoi path mentioned above):

- *BASEDIR*/smarts/conf/trapd (where *BASEDIR* may be either InCharge/SAM/ or InCharge/IP/)

  Be sure to use the *sm_edit* utility to edit the seed file so that passwords will be encrypted.

  Verify that the secret phrase used to re-encrypt the seed file matches for all products receiving traps from the SNMP Trap Adapter. The System Administration Guide provides more information about configuring the secret phrase used to encrypt seed files.

  Note Although the SNMP Trap Adapter can parse plain text passwords from unencrypted seed files, VMware strongly recommends that all seed files be encrypted by using the *sm_edit* utility. The System Administration Guide provides more information.

Editing seed files

The format of the seed file is identical to a standard IP Domain Manager seed file, but the semantics are slightly different. In IP, the sections are keyed by the hostname or IP address (the first line in each block). SNMPv3/USM uses the engineID of the SNMP agent in the network device to uniquely identify the agent which sent the trap. The hostname (or IP address) is ignored completely. For each engineID, there will be one or more users. The engineID/userName pair constitutes the unique key for the SNMP Trap Adapter seed file.

Note To receive SNMPv3 traps, you must uncomment and edit specific lines. For example, you must type a valid hostname or IP address, set the SNMPVERSION field to V3, and edit the seed file fields beginning with the comment: # The following are for SNMPv3 entries. Recommended practice is to place each SNMPv3 field/value pair on its own line. Sample SNMPv3 seed file entries provides more information.

The first line in the seed file must appear as follows if you intend to encrypt both the AUTHPROTOCOL and PRIVPROTOCOL passwords:

```
#<encrypted seed>:1.0:AUTHPASS,PRIVPASS
```

When you edit the seed file by using the **sm_edit** utility, this line controls which field values in the seed file should be encrypted (for example, the AUTHPASS and PRIVPASS password fields).

The **sm_edit** utility may be invoked in a non-interactive mode by using the **noedit** option, for example:

```
sm_edit --noedit conf/trapd/seedfile
```

This will cause *sm_edit* to read the seed file, encrypt the fields specified by the first line, and write them back in an encrypted mode.

Note While *sm_trapd* can parse plain text passwords from unencrypted seed files, VMware strongly recommends that all seed files be encrypted by using the *sm_edit* utility. Failure to encrypt the seed files (and destroy any plaintext intermediates) will expose plaintext passwords to anybody who can read the file.

Sample SNMPv3 seed file entries

```
#<encrypted seed>:1.0:AUTHPASS,PRIVPASS
128.221.19.8
SNMPVERSION=V3
USER=shaAESUser
AUTHPROTOCOL=SHA
AUTHPASS=123ABC456#$%123abc456#$%
PRIVPROTOCOL=AES
PRIVPASS=456123abc456#$%456123ABC456#$%
ENGINEID=0000000902000003E333C440
qa-gwipv6
SNMPVERSION=V3
```

```
USER=MD5DesUser
AUTHPROTOCOL=MD5
AUTHPASS=789ABC456#$%789abc456#$%
PRIVPROTOCOL=DES
PRIVPASS=789123abc456#$%789123ABC456#$%
```

ENGINEID=000000090200000F134B93C

SNMPv3-related seed file field descriptions lists and describes seed file fields for SNMPv3.

Table 3-1. SNMPv3-related seed file field descriptions

Field	Description
AUTHPROTOCOL	Specifies the authentication protocol in use by the network device sending the SNMPv3 traps. Valid entries include MD5, NONE, or SHA. For the SNMP trap processor, if no value is specified, it defaults to NONE. **Note** The default value used in the discovery process by the IP Manager may differ. The *IP Manager Release Notes* provide more information.
AUTHPASS	Authentication password. This may be 64 characters long. VMware, Inc. recommends the use of complex passwords (for example, a long string of uppercase or lowercase alpha characters, numbers, and special characters)
ENGINEID	An engineID is a unique string identifying an SNMP engine. It does not uniquely identify a DEVICE, because technically, a device can host multiple SNMP agents. So, the engine ID is assigned to each agent, not device. If you have IP Availability Manager running, you can use the sm_tpmgr -s <*server name*> **--dump-agents** command to find the engineIDs for those agents generating SNMPv3 traps in your network.
PRIVPASS	Privacy (encryption) password. This may be 64 characters long. VMware, Inc. recommends the use of complex passwords (for example, a long string of uppercase or lowercase alpha characters, numbers, and special characters).
PRIVPROTOCOL	Privacy (encryption) protocol in use by the router sending the SNMPv3 traps. Valid entries include DES, NONE, or AES. If no entry, defaults to NONE.

To simplify debugging of your network device configurations, VMware, Inc. recommends that you first establish that SNMPv3 traps sent from the network device in *noAuthNoPriv* mode are processed correctly. When you are sure that it works, use *authNoPriv* and finally *authPriv*, after verifying correct operation at each step.

Encryption (Privacy) options supported in SNMPv3 seed file

The following seed file options are used to encrypt SNMPv3 traps coming from the network to the SNMP trap exploder (SNMP Trap Adapter):

- PRIVPASS
- PRIVPROTOCOL

  The IP Manager User Guide and the System Administration Guide provide additional information.

To enable the SNMPV3 Privacy Protocol and Privacy Password fields located in the Add Agent dialog box:

a From the **Add Agent** dialog box, click **Advanced Options**.

The **Advanced Options** pane appears.

b Select **V3** from the **SNMP Version** field.

The **SNMP V3 Specifications** pane appears.

c Select a privacy protocol from the **Privacy Protocol** field.

d Type a password in the **Privacy Password** field.

e Click **OK**.

Load the seed file into the Local Credentials Database (LCD)

Provide agent credentials to the SNMP Trap Adapter by loading one or more seed files using the *importSeedFile.asl* script. This script writes the content of the seed file to the LCD.

The sample startup script given above will cause the SNMP Trap Adapter to read in a pre-existing seed file when the program begins execution. This is most useful in a test environment, or in a fairly small and static installation. In a large network, it is expected that new devices will be added, and user credentials changed, on a regular basis. To accommodate this, the SNMP Trap Adapter can read seed files while it is running.

To read a new seed file, invoke the ASL script, *importSeedFile.asl*, and give a single command line option specifying the name of the seed file:

1 Go to *BASEDIR/smarts/bin* for the Service Assurance Manager product.

2 Type the following command:

```
sm_adapter -s TRAP-INCHARGE-OI -D seed=seedfile trapd/importSeedFile.asl s
```

The seed file will be parsed, and the new USM credential data will be merged into the existing LCD.

Managing seed file updates

The seed file entries used by the SNMP Trap Adapter are keyed by engineID and userName pair. Any entry which has the same engineID and userName pair as existing data in the LCD will result in the new data overwriting the old.

In some cases, it may be convenient to split the USM credential data into several seed files (for example, one seed file corresponding to each of several IP domain managers). You may run *importSeedFile.asl* after for each seed file; and they will be read in turn. All of the data from all of the seed files is merged into a single LCD. If there are duplicate engineID and userName pairs between files, the last one read is the one which will be kept.

There is currently no way to delete a user from the LCD. In many instances, having obsolete user data in the LCD will cause no operational harm, and old entries can simply be ignored. If you want, you may import a seed file containing the obsolete engineID and userName pair with a non-matching password to effectively disable that entry.

Configuring trapd.conf (trap exploder and trap receiver)

To configure the trap receiver to listen for traps, edit the trapd.conf file. There are multiple trapd.conf files includes the products. The Parameters in trapd.conf file table lists and describes the parameters in the trapd.conf file.

To determine which trapd.conf file to edit depends upon your deployment scenario:

- For Scenario 1 (single trap receiver), edit InCharge/SAM/smarts/conf/icoi/trapd.conf. For example, to forward traps to the IP Availability Manager, remove the # to uncomment the FORWARD line:

```
# Traps required by InCharge IP Availability Manager (AM)
#
# Generic: coldStart, warmStart, LinkUp, LinkDown
# FORWARD: *.*.*.* .* <0-3> * host:port
For example:
FORWARD: *.*.*.* .* <0-3> * AM_HOST-NAME:AM-PORT
```

- For Scenario 2 (one trap exploder and one trap receiver), edit:

```
               BASEDIR/smarts/conf/trapd/trapd.conf
FORWARD: *.*.*.* .* * * TRAP-ADAPTER_HOST-NAME:TRAP-ADAPTER-PORT
FORWARD: *.*.*.* .* <0-3> * AM_HOST-NAME:AM-PORT
```

Parameters in trapd.conf file

The trapd.conf file is located under:

- For Trap Adapter: <**BASEDIR**>/smarts/conf/trapd directory
- For Trap Exploder: <**BASEDIR**>/smarts/conf/icoi directory

  You need to use sm_edit to edit the parameters in the trapd.conf file. Parameters in trapd.conf file lists and describes the parameters in the trapd.conf file.

Table 3-2. Parameters in trapd.conf file

Configuration file parameter	Description	Valid values	Applicable for Adapter or Exploder
PORT	The UDP port number the trap adapter listens to. If NetView or OpenView is running on this system, during installation, the value is set to 9000, otherwise it is set to 162	0-65535	Adapter
WINDOW	This is the de-duplication window, in seconds. The maximum amount of time between receiving similar traps, before the second trap is considered unique. To disable de-duplication feature make all traps unique. That is, do not specify a value for this parameter or use zero. If a value is not specified for this parameter, or it is set to zero, de-duplication is disabled.	Non-negative integer	Adapter
THREADS	The number of trap processing threads to spawn. The determines how many traps can be processed concurrently. The maximum value is 25. If not set, the default value is 1.	0 - 25	Adapter and Exploder
ASCII	This controls the formatting of non-printable characters. If the value for ASCII is set to FALSE or if “--ascii” is not specified on the commandline for starting sm_trapd, sm_trapd converts the entire value of that OID into a printable UTF-8 coding string, such as a varbind. For example, .1.3.6.1.4.1.333.1 -> "abcd012" is converted into .1.3.6.1.4.1.333.1 -> "61 62 63 64 C3 96 30 31 32". If the value for this parameter is set to TRUE or if “--ascii” is specified on the commandline for starting sm_trapd, sm_trapd replaces all the non-printable characters with an “X” followed by the HEX string of these characters using UTF coding. The remaining printable characters of the original value (octet-string) remain unchanged, such as the varbind. For example, .1.3.6.1.4.1.333.1 -> "abcd012" is then converted into .1.3.6.1.4.1.333.1 -> "abcdXC3X96012."	TRUE | FALSE	Adapter and Exploder
SOURCE	This switch enables the trap processor to obtain the source address of the IP packet, and makes it available to the customer configurable .conf file and .asl scripts. If not set, value is FALSE.	TRUE | FALSE	Adapter and Exploder
TAG	Enables tagging of varbind values. When enabled, it streams the value's type before each value. For example, INTEGER-32 3.	TRUE | FALSE	Adapter and Exploder

Table 3-2. Parameters in trapd.conf file (continued)

Configuration file parameter	Description	Valid values	Applicable for Adapter or Exploder
QUEUE_LIMIT_MEGS	This parameter helps limit the size of internal trap queue to the stated size. The limit is not exact - the queue may grow slightly larger than that. When the limit is reached, some traps will be discarded. The default value is set to 200 MB. On reaching this limit, the sm_trapd queue starts discarding traps. When the queue starts discarding traps, a *trapdDiscardingTraps* alert is generated, as defined in the trap_mgr.conf file. If and when a device is identified as the source of a trap storm, information about it is logged in the sm_trapd log file. The alert is cleared once the incoming trap rate to the sm_trapd reduces to a rate below which these traps can be processed by the queue.	Non-negative integer	Adapter and Exploder
QUEUE_LIMIT_SECONDS	Limit the time that a trap can remain in the internal trap queue. This limit is even less exact than the size limit above. In general, it is advisable to specify both. When traps start to remain in the queue exceeding the time limit set, some traps will be discarded. The default value is set to 480 seconds. When the queue starts discarding traps, a *trapdDiscardingTraps* alert is generated, as defined in the trap_mgr.conf file. If and when a particular device is identified as the source of a trap storm, information about it is logged in the sm_trapd log file. The alert is cleared once the incoming trap rate to the sm_trapd reduces to a rate below which these traps can be processed by the queue.	Non-negative integer	Adapter and Exploder
ENABLE_FWD	This parameter enables trap forwarding. Traps are only forwarded if a forwarding criteria is specified. See the FORWARD parameter for more information.	TRUE | FALSE	Exploder
MATCH	Determines whether traps are tested against all forwarding criteria or up to the first criterion that matches. If forwarding criteria is not specified, this parameter is ignored. Forwarding criteria is specified in this file using the FORWARD parameter. If not set, value is first.	all | first	Exploder

Table 3-2. Parameters in trapd.conf file (continued)

Configuration file parameter	Description	Valid values	Applicable for Adapter or Exploder
TIMESTAMP_RCV	When receiving a trap, whether to forward the time from the trap to ASL as the received time of the trap. The original behavior of the trap processor received timestamps of this format and this option allows that behavior. If set to FALSE this defaults to sending the timestamp from the trap itself which is in normal SNMP time format of hundredths of a second since device boot.	TRUE | FALSE	Adapter and Exploder
FORWARD	Specifies the matching criteria for incoming traps and the forwarding destinations for matched traps. Valid syntax is: <source device address> <OID> <generic type> <specific type> \ <destination host address>[:<port>|:<port>:<community>] \ [<destination host address>[:<port>|:<port>:<community>]] ... where: ■ <*source device address*> is the IP address (IPv4, IPv6) of the object (SNMP agent) that is generating the trap. ■ <*OID*> is the sysObjectID of the type of object that is generating the trap. ■ <*generic type*> is the generic trap type: 0 coldStart 1 warmStart 2 linkDown 3 linkUp 4 authenticationFailure 5 egpNeighborLoss 6 enterpriseSpecific Valid syntax for <*generic type*> is a generic specific trap number (for example, 3), a range of generic specific trap numbers (for example, <3-5>), or any generic specific trap number (for example, *). An asterisk is a wildcard character that matches any arbitrary string of characters. ■ <*specific type*> is the specific trap code, present even if <*generic type*> is not enterpriseSpecific (6). Valid syntax for <*specific type*> is an enterprise specific trap number (for example, 733), a range of enterprise specific trap numbers (for example, <130-156>), or any enterprise specific trap number (for example, *).		

Globbing can be used to specify values of the following fields: <address>, <OID>, <generic type>, <specific type>. Field description lists and describes the fields:.

Table 3-3. Field description

Field name	Description
<address>	Source IP address or host name of the SNMP agent sending the trap. Globbing can be used to specify ranges of IP addresses. Examples of valid values: 192.168.114.5 3ffe:80c0:22c:109:214:4fff:fe39:a73d *.*.*.* *:*:* 192.168.*.* 3ffe:80c0:22c:* *.*.*.168 *:*:a73d 192.168.<120-123>.* 3ffe:80c0:22c:109:214:4fff:<0-ffff>:*
<OID>	Enterprise OID of incoming trap. Globbing can be used to specify ranges of OIDs. Examples of valid values: .1.3.6.4.1.9 .1.3.* .* .1.3.6.4.1.<1-18>.*
<generic type>	Generic type of incoming trap. Globbing can be used to specify ranges of trap numbers. Examples of valid values: 0, 3, <0-5>
<specific type>	Specific type of incoming trap, if valid. For standard (generic) traps this field is ignored. Globbing can be used to specify ranges of trap numbers. Examples of valid values: 733, <130-156>, *
<host <[:port] | [:port:community]>>	Destination to forward matched traps to. Multiple destinations can be specified. Host is specified as an IP address or a host name. Port and Community are optional; IPv6 addresses need to be enclosed in square brackets [] if omitted, port 162 is used and the community string of the incoming trap is preserved. Globbing cannot be used in this field. Examples of valid values: cobra, snake.planet.net:6789, 192.76.70.21, [3ffe:80c0:22c:109:214:4fff:fe39:a73d]:2002, 192.168.70.190:6789

Some more examples:

- All traps from all IPv4 addresses are forwarded to IPv6 host gifted.yahoo.com on port 2002:

```
FORWARD: *.*.*.* .* * * gifted.yahoo.com:v6:2002
```

- All traps from all IPv6 addresses are forwarded to IPv6 host gifted.yahoo.com on port 2002:

```
FORWARD: *:*:* .* * * gifted.yahoo.com:v6:2002
```

- All traps from all IPv6 addresses are forwarded to IPv6 host with address 3ffe:80c0:22c:109:214:4fff:fe39:a73d on port 2002:

```
FORWARD: *:*:* .* * * [3ffe:80c0:22c:109:214:4fff:fe39:a73d]:2002
```

- All traps from all IPv4 addresses are forwarded to IPv6 host gifted.yahoo.com on port 2002, smarts OIDs smTrapAddress and smTrapAddressType are not added when forwarding.

```
FORWARD: *.*.*.* .* * * -nosmtrapaddr gifted.yahoo.com:v6:2002
```

- All traps from all IPv6 addresses are forwarded to IPv6 host gifted.yahoo.com on port 2002, smarts OIDs smTrapAddress and smTrapAddressType are not added when forwarding.

```
FORWARD: *:*:* .* * * -nosmtrapaddr gifted.yahoo.com:v6:2002
```

- All traps from all IPv6 addresses are forwarded to IPv6 host with address 3ffe:80c0:22c:109:214:4fff:fe39:a73d on port 2002, smarts OIDs smTrapAddress and smTrapAddressType are not added when forwarding.

```
FORWARD: *:*:* .* * * -nosmtrapaddr [3ffe:80c0:22c:109:214:4fff:fe39:a73d]:2002
```

- All traps from all IPv4 addresses are forwarded to IPv4 host amtest.smarts.com (on the default SMTP port) and IPv6 host gifted.yahoo.com (on port 2002). MIB variables smTrapAddress and smTrapAddressType are added when forwarding to amtest.smarts.com (default behavior). The same OIDs are not added when forwarding to gifted.yahoo.com, due to the -nosmtrapaddr option that precedes this destination entry.

```
FORWARD: *.*.*.* .* * * ipam.smarts.com:v4:2002 -nosmtrapaddr gifted.yahoo.com:v6:2002
```

Configuring the SNMP Trap Adapter to forward notifications

SNMP Trap Adapter configuration and script files describes the files that you use to configure the SNMP Trap Adapter to forward notifications on to the Global Manager.

Note You can define up to 25 trap processing threads in your local trapd.conf file, which includes a definition and default value for THREADS. Running a multi-threaded trap adapter may eliminate the need to send these traps to a forwarder.

To configure the SNMP Trap Adapter, complete the following steps:

1 Define the traps that you want to forward to the Global Manager in your local copy of trap_mgr.conf file.

2 Set up the proper security between the Adapter Platform and the SNMP Trap Adapter. SNMP Trap Adapter security provides more information.

3 Verify the port setting for the SNMP Trap Adapter. Verifying the SNMP Trap Adapter port setting provides more information.

4 Configure the SNMP Trap Adapter to start and stop automatically.

5 Start the SNMP Trap Adapter. Starting and stopping the SNMP Trap Adapter provides more information.

Table 3-4. SNMP Trap Adapter configuration and script files

Directory under *BASEDIR/*	Filename	User editable	Description
smarts/conf/icoi	trap_mgr.conf	Yes	Used to map incoming SNMP traps to notifications, and to configure notification batch parameters.

Overview of trap_mgr.conf

The trap receiver uses the trap_mgr.conf configuration file to translate incoming traps into notifications that are sent to the Global Manager. You need to use **sm_edit** to edit trap_mgr.conf (found at *BASEDIR*/smarts/conf/icoi) to define the traps that you want to send to the Global Manager. For situations requiring more advanced trap processing, an ASL script can be called for specific traps.

The trap_mgr.conf file contains several sections:

- The beginning of the file includes comments regarding the syntax of the file.
- The first line of code sets the BATCH_NOTIFY_INTERVAL. SNMP trap batching provides more information about this interval.
- A default section, which lists default values for the fields. If, for a given trap, no value is set for a field, the value from the default section is used. This section begins with BEGIN_DEFAULTS and ends with END_DEFAULTS.
- Trap definition sections, which define mappings of information for specific traps or groups of traps. There can be one or more trap definition sections. Each section begins with BEGIN_TRAP and specifies the trap mapping by OID, generic trap number, and specific trap number. Each trap definition ends with END_TRAP.

  Default trap parameter section on page 61 provides more information. Chapter 7 VMware Smart Assurance MIB for SNMP Traps includes the MIB implemented by VMware, Inc. for SNMP traps.

SNMP trap example

Translation of SNMP trap to standard notification shows an example of a simple trap and how values in the trap might populate fields in a Service Assurance notification. In this example, the SNMP trap consists of five values:

- Enterprise OID
- Generic Trap Number
- Specific Trap Number

- System Name

  The system name sent by the trap translates into the instance name.

- Varbind 1

  Varbind 1 translates into a text message for the notification.

  Other fields in the notification get populated from values placed in the configuration file. For example, the ClassName is Host not because the information was sent with the trap, but because an association was made in the configuration file.

  .1.3.6.1.4.1.2.1

  0

  0

  HostNYC8

  Operator requested reboot

  SNMP Trap (simplified)

  .1.3.6.1.4.1.2.1

  0

  0

  HostNYC8

  Operator requested reboot

  Enterprise OID

  Generic Trap Number

  Specific Trap Number

  System Name

  Varbind 1

  Incoming SNMP Trap

  Cold Start

  HostNYC8

  InstanceName

  EventName

  Host

  ClassName

  EventType

  MOMENTARY

EventText

Agent HostNYC restarted, reason=Operator requested reboot

Severity

5

Standard Notification Fields

For this example, no translation into any standard notification fields. These fields are used as keys to the configuration file.

Figure 3-1. Translation of SNMP trap to standard notification

The following is an example of a portion of the configuration file used by the SNMP Trap Adapter. This example corresponds to Translation of SNMP trap to standard notification. When the trap matches a trap definition (Enterprise OID, generic trap number, and specific trap number), the Adapter Platform populates attributes in a notification based on entries in the trap definition. In the example, the standard notification attribute ClassName is populated with the value Host:

```
BEGIN_TRAP  .1.3.6.1.4.1.2.1  0  0
ClassName:   Host
InstanceName:   $SYS$
EventName:   Cold Start
Severity:   5
EventType:   MOMENTARY
EventText:   Agent $SYS$ restarted, reason=$V1$
END_TRAP
```

Default trap parameter section

The format of the default section is as follows:

```
BEGIN_DEFAULTS
 ClassName:   SNMPTrap
 InstanceName:   $SYS$
 EventName:   $E$ $N$ $S$
 Severity:   2
 EventText:   Varbinds: $V*$
 Expiration:   7200
 State:   NOTIFY
 InMaintenance:   FALSE
 ClearOnAcknowledge:   TRUE
 EventType:   MOMENTARY
 SysNameOrAddr:   $A$
 UnknownAgent:   IGNORE
 LogFile:   NONE
END_DEFAULTS
```

The default section demonstrates that you can use variables in descriptions. Of particular note is the use of the varbind variable with an asterisk for the EventText parameter. In this case, EventText stores the value of all of the varbinds associated with a trap. By defining LogFile, the SNMP Trap Adapter logs all of the traps unless a trap's LogFile is defined as NONE.

Trap definition sections

The configuration file contains multiple trap definitions. Each definition specifies what set of traps are processed by the trap definition and what notification values are set for those traps. Incoming SNMP traps are matched against trap definitions in a configuration file.

The beginning of each trap definition starts with BEGIN_TRAP. Each incoming trap is identified by three fields following BEGIN_TRAP. These fields are: Enterprise OID, generic trap number, and specific trap number. The format of the first line is:

```
BEGIN_TRAP <enterprise> <generic_trap> <specific_trap>
```

Each of the fields need to be separated by either a space or a tab. The first field contains the Enterprise OID. The first six numeric values in this field are expected to conform to either the standard Enterprise (vendor) prefix of 1.3.6.1.4.1 or the standard MIB-II (generic) prefix of 1.3.6.1.2.1. However, the SNMP Trap Adapter does not restrict you to these numeric values. The first field can be any valid OID number or the string 'any' (or *).

You can specify a range or use wildcards in the range. For example, you can specify a range as .1.3.6.1.4.1.<*4-10*> where the last numeric value can be within the range of 4 through 10. In the example 1.3.6.1.4.1.*, the "*" character matches any number for the last OID numeric value.

Additionally, you can list multiple OIDs at the beginning of each trap definition. Specifying multiple OIDs allows the same trap definition to be used for multiple traps. The following example demonstrates listing multiple OIDS, specifying a numeric value range, and using a wildcard:

```
BEGIN_TRAP .1.3.6.1.4.1.1.2.<23-40> 6 1 .1.3.6.1.4.1.6.* 6 1
```

Trap definitions are read from left to right. For example, suppose a configuration file includes multiple trap definitions such as:

```
BEGIN TRAP .1.2.3.4.5.6.*   *  2
BEGIN TRAP .1.2.3.4.5.6.*   1  *
```

Your system receives a trap with the definition:

```
BEGIN TRAP .1.2.3.4.5.6.*   1  2
```

The trap matches both trap definitions included in the configuration file. However, because traps definitions are read from left to right, the second trap definition is used for the trap.

SNMP trap parameters

The trap definition follows the BEGIN_TRAP line. The trap definition is the set of values for specified notification attributes that are placed in the Service Assurance notification. You can define these values in any order. The syntax of a line from the trap definition is:

```
<notification_attribute>:  <value>
```

Trap definition parameters shows the complete list of parameters to use in a trap definition and in the BEGIN_TRAP line. Each trap definition ends with END_TRAP.

Note If no parameter values are set within a trap definition BEGIN_TRAP section, the values set in the BEGIN_DEFAULT section are used.

Table 3-5. Trap definition parameters

Configuration file parameter	Corresponding standard notification attribute or description	Valid values
ASL	For advanced SNMP trap integration, this defines the ASL rule set file used to perform additional processing, which may include additional variable substitution. The ASL rule set must be located in *BASEDIR*/smarts/local/rules/icoi_trapd. (A sample rule set is available in my_trap-rules.asl.) The ASL processing overwrites any values set in the trap definition section.	Filename
Aggregate	For advanced SNMP trap integration, this defines the mapping of a trap definition to an aggregate notification. One or more component notifications comprise an aggregate notification. Using the Trap Adapter Aggregate parameter provides details and examples. **Note** The Aggregate field is not supported in the Default Section of the file.	Special ■ EventName:<*string*> This is the name of the aggregate notification and is required. ■ ElementName:<*object-handle*> An object-handle identifies the InstanceName and the CreationClassName of an element. This is the name of the topology element where the aggregate is defined. This field is optional. If this field is not defined, the ElementClassName and ElementName of the trap definition are used, if available. Otherwise, the ClassName and InstanceName of the trap definition are used. If you define this field and specify an object-handle that does not exist in the topology, then the aggregate is not created. ■ EventText:<*string*> This is the description of the aggregate notification (event). This field is optional.

Table 3-5. Trap definition parameters (continued)

Configuration file parameter	Corresponding standard notification attribute or description	Valid values
Category	Category	Any string
ClassName	ClassName and DisplayClassName	Any string
ClearOnAcknowledge	ClearOnAcknowledge	TRUE or FALSE
Discard	Discard the event information captured by the trap: ■ YES discards event information. ■ NO saves event information. ■ IF_MANAGED allows a discard of the event information if the device is managed by another connected server, thus avoiding possible duplicate event information from both servers.	YES, NO, or IF_MANAGED **Note** IF_MANAGED requires that the ICOI server subscribes to all underlying IP servers for consideration to determine if the device that originated the trap is managed.
ElementClassName	ElementClassName	Valid Class Name in topology.
ElementName	ElementName	Valid Instance Name in topology. If you set the UnknownAgent to CREATE, then the instance does not have to previously exist. **Note** If the UnknownAgent parameter is configured to CREATE, the ElementClassName and ElementName parameters must be defined in the trap_mgr.conf file.
EventName	EventName (describing the notification) and the EventDisplayName	Any string
EventText	EventText	Any string
EventType	EventType: ■ MOMENTARY sends notifications of events with no duration, such as a coldStart trap. Clears after the Expiration interval. ■ DURABLE sends notifications of events with an active and inactive state, such as linkDown and linkUP traps.	MOMENTARY or DURABLE
Expiration	Defines the expiration time in seconds for the notification. Zero indicates that the notification will not expire. 7200 is the default value. If you use zero, you should use some method to eventually clear the notification. For example, you can configure the SNMP Trap Adapter to send clear state notifications or set the ClearOnAcknowlege parameter to TRUE.	Integer

Table 3-5. Trap definition parameters (continued)

Configuration file parameter	Corresponding standard notification attribute or description	Valid values
ForceOccurredOn	Overwrite the ElementClass or ElementName to modify the OccurredOn object. **Note** If ElementClass and ElementName are undefined, the trap defaults to the Agent name if it is defined.	TRUE or FALSE
InMaintenance	InMaintenance	TRUE or FALSE
InstanceName	InstanceName (of the object associated with the notification) and InstanceDisplayName.	Any string
LogFile	The name of the file used by the SNMP Trap Adapter to log information for this trap. If this parameter is NONE or undefined, no information is logged. If this parameter is undefined, the default value is used. **Note** Log files contain any error messages with line numbers of the trap_mgr.conf to help you find and correct the error.	Filename
Map	Map is a special field that enables you to map varbinds to one or more printable strings. Using the Map parameter and Using the Map parameter with tags provide more information and examples of how to map varbind values. **Note** The Map field is not supported in the Default Section of the file.	Special
Severity	Severity	Any integer from 1 to 5
State	Describes the state of the notification.	NOTIFY or CLEAR
SysNameOrAddr	A valid value describes either the system name or address of the entity that sent the trap. Values for this parameter override values in ElementClassName and ElementName.	Any string
TrapSource	Define the name of your trap processor so that it more closely identifies the function or form of your traps. The default name isTrap Processor.	Any string
UnknownAgent	Describes whether Adapter Platform should ignore traps related to unknown topology elements or create elements for the traps.	CREATE or IGNORE **Note** If the UnknownAgent parameter is configured to CREATE, the ElementClassName and ElementName parameters must be defined in the trap_mgr.conf file.

Table 3-5. Trap definition parameters (continued)

Configuration file parameter	Corresponding standard notification attribute or description	Valid values
UpdateUD	Update the user-defined trap and data with a CLEAR.	TRUE or FALSE
UserDefined1-10	UserDefined1-10	Any string

In the SNMP Trap parametes table and the corresponding trap definition, certain values in the SNMP trap were used as attribute values in the notification. This is accomplished through the use of variable substitution. SNMP variable bindings (varbinds) are placeholders for information common to standard SNMP traps and can be assigned to Service Assurance notification attributes within the trap definitions in the trap_mgr.conf file. The following two lines are taken from the trap definition based on SNMP Trap parametes that use variable substitution:

```
InstanceName:$SYS$
EventText:Agent $SYS$ restarted, reason=$V1$
```

The attribute InstanceName is populated with the value of a variable SYS. The contents of this variable come from the incoming trap. In this example, SYS is the name of the system where the trap originated, and $V1$ represents the value of the first varbind. The SNMP Trap Adapter variables describes the variables available for basic SNMP trap integration. When specifying a variable, it must be enclosed within dollar signs ($).

Table 3-6. SNMP Trap Adapter variables

Variable	Description
A	Address of the agent sending the trap.
C	Community string of the SNMP trap. **Note** To use this variable, you must start the trap adapter with the **--community** option.
E	Enterprise OID of the SNMP trap.
N	Generic trap number of the SNMP trap.
S	Specific trap number of the SNMP trap.
SRC	Source, or originator, of the IP address for the trap. A V1 trap may return the source IP address of the relay agent for the trap. **Note** To use this variable, you must specify the --source parameter.
SYS	System where the SNMP trap originated.

Table 3-6. SNMP Trap Adapter variables (continued)

Variable	Description
T	Timestamp of the SNMP trap.
$V<*n*>[-<*tag*>]$ or $OID<*n*>[-<*tag*>]$ or $V<*>$ or $OID<*>$	Value of the <*n*>th varbind. The varbind may contain one of the following data types: integer, bit-string, octet-string, IP address, counter, gauge, unsigned integer, time ticks, counter 64, obj id, opaque, or null. Optionally, you can use a text string to tag the value of the <*n*>th varbind for multiple mappings. You can also use an asterisk <*> to specify all varbinds.

Example of set and clear notification

The following example illustrates how to configure the notification and clear an event. One trap creates a state of NOTIFY, while the other creates a state of CLEAR:

```
BEGIN_TRAP  .1.3.6.1.4.1.10.1.9.5.1  6  1
ClassName:   Port
InstanceName:   $SYS$ PORT $V1$
EventName:   lerAlarmOn
Severity:   3
EventText:   This is a longer text message
EventType:   DURABLE
Category:   Performance
State:   NOTIFY
END_TRAP
BEGIN_TRAP  .1.3.6.1.4.1.10.1.9.5.1  6  2
ClassName:   Port
InstanceName:   $SYS$ PORT $V1$
EventName:   lerAlarmOn
State:   CLEAR
END_TRAP
```

Notifications are uniquely identified by using the three attributes: ClassName, InstanceName, and EventName. In the example, the second trap definition will clear an event created with the first trap definition when the values of the three key fields are identical.

Using the Map parameter

The format of the Map parameter is:

```
Map:  {
  V<n>
   <value_a>= <string>
   <value_b>= <string>
  }
  {
  V<n>
```

```
    <value_c>= <string>
    <value_d>= <string>
   }
```

The following example shows how the Map parameter could be used to substitute text for two different varbind values:

```
BEGIN_TRAP .1.3.6.1.4.1.10 6 0
 ClassName:    Host
 InstanceName:    $SYS$
 EventName:    Coldstart
 Severity:    5
 EventText:    Reason - $V1$
 Expiration:    30
 State:    NOTIFY
 UserDefined1:    Community String: $C$
 ClearOnAcknowledge:    TRUE
 Map:    {
     V1
      1= UP
      2= DOWN
     }
 EventType:    DURABLE
 UnknownAgent:    CREATE
 ElementClass:    Router
 ElementInstance:    $SYS$
 LogFile:    Coldstart.log
END_TRAP
```

In the example, the Map parameter describes how the enumeration values of the first varbind translate into printable strings. These printable strings are substituted as values in EventText. If a SNMP trap had a value of one for its first varbind, the phrase “Reason - UP” would be placed in EventText.

If a map is not defined for a specific varbind, the SNMP Trap Adapter uses the varbind’s integer value. In the example, the integer “1” would be used instead of the text “UP,” so the phrase “Reason -1” would be placed in EventText.

Note For a trap to clear, the InstanceName and ElementInstance parameters must be set to the same type.

Using the Map parameter with tags

Each incoming trap is identified by three fields: Enterprise OID, generic trap number, and specific trap number. Some SNMP agents send traps with the same combination of OID, generic, and specific trap numbers but for different reasons. When this occurs, the only way to differentiate the meaning of the trap is to examine the varbinds. You can use the $V<*n*>-[<*tag*>]$ variable to map varbind values to different text strings.

The format of the Map parameter when using tags is as follows:

```
Map:  {
  V<n-tag1>
   <value_a>= <string1>
   <value_b>= <string2>
   default= <string3>
  }
  {
  V<n-tag2>
   <value_a>= <string4>
   <value_b>= <string5>
   default=
  }
  }
  V<n-tag3>
   <value_a>= <string6>
   <value_b>= <string7>
   default=
```

The following example defines two $V<*n*>-[<*tag*>]$ variables, one for substitution of ClassName and another for substitution of EventName:

```
BEGIN_TRAP .1.3.6.1.4.1.546.1.1 6
 ClassName:    $V2-class$
 EventName:    $V2-event$
.
.
.
```

The mapping defines when to map the appropriate tags to the varbinds.

```
.
.
.
 Map:  {
   V2-class
   .1.3.6.1.4.1.546.1.1.7.9.30.0= Memory
   .1.3.6.1.4.1.546.1.1.7.9.2.0= Processor
   default=Host
   }
   {
   V2-event
   .1.3.6.1.4.1.546.1.1.7.9.30.0= NotEnoughMem
   .1.3.6.1.4.1.546.1.1.7.9.2.0= HighUtilization
   default= TrapReceived
   }
```

If the incoming trap has a Varbind 2 value of .1.3.6.1.4.1.546.1.1.7.9.30.0, then Memory is used for the ClassName and NotEnoughMem is used for the EventName. If the incoming trap has a Varbind 2 value of .1.3.6.1.4.1.546.1.1.7.9.2.0, then Processor is used for the ClassName and HighUtilization is used for the EventName. The default, Host, is used for the ClassName and the default, TrapReceived, is used for the EventName for incoming traps with any other Varbind 2 values. If a default is not specified, the original value for the Varbind 2 is returned.

Using the Trap Adapter Aggregate parameter

An aggregate notification is a notification that is composed of one or more component events. When one of the component events occurs, the aggregate notification is notified. Creating an aggregate notification provides several benefits:

- Aggregation compresses multiple component events into a single aggregate notification.
- Aggregation helps you organize related traps into meaningful categories.
- Aggregation provides a method by which the details of the component events can be displayed to an operator. The component events of an aggregate notification are displayed in the Aggregate tab of the Notification Properties window of the Global Console.

  You use the aggregate in your trap definition to create an aggregate notification as well as the aggregate's component events. You are required to define an EventName for each aggregate or component event. Because all notifications are uniquely identified by EventName and the ClassName, and the InstanceName it OccurredOn, values must also be derived for the two latter attributes. VMware, Inc. recommends that you explicitly define the ElementName:<*object-handle*> within the Aggregate portion of the trap definition. If you do not explicitly define the ElementName within the Aggregate portion, then the ElementClassName and ElementName of the trap definition are used, if available. Alternatively, if ElementName and ElementClassName are not defined, then the ClassName and InstanceName are used to define the aggregate.

  The following example illustrates how to configure two different trap definitions to generate the same aggregate notification. The first trap definition processes an incoming trap that indicates a system is experiencing low memory. The second trap definition processes an incoming trap that indicates that a system's processor is experiencing high utilization. As both traps indicate system degradation, if either trap is received, a notification is generated to indicate that the system is degraded:

```
BEGIN_TRAP .1.3.6.1.4.1.546.1.1 6 1
 ClassName:    Memory
 InstanceName:    MEM-$SYS$
 EventName:    Low
 Aggregate:    {
       EventName: Degraded
       ElementName:$SYS$

     }
 Severity:    1
 Expiratiton:    30
```

```
 State:    NOTIFY
 ClearOnAcknowledge:    TRUE
END_TRAP
BEGIN_TRAP .1.3.6.1.4.1.546.1.1 6 2
 ClassName:    Processor
 InstanceName:    PRO-$SYS$
 EventName:    HighUtilization
 Aggregate:    {
       EventName: Degraded
       ElementName:$SYS$

     }
 Severity:    1
 Expiratiton:    30
 State:    NOTIFY
 ClearOnAcknowledge:    TRUE
END_TRAP
```

Using the example, if a trap with OID .1.3.6.1.4.1.546.1.1 6 1 is received from Router::CoreRouter1, the component event Memory::MEM-CoreRouter1 Low is generated. Similarly, if a trap with OID .1.3.6.1.4.1.546.1.1 6 2 is also received from the same router, then the component event Processor::PRO-CoreRouter1 HighUtilization is generated. Because the same aggregate is defined for both of these traps, this will also generate the aggregate notification, Router::CoreRouter1 Degraded.

Although all events are propagated to the Service Assurance Global Manager, only the aggregate notification (Router::CoreRouter1 Degraded) will appear in the Global Console. This is because the default NotificationList used by the Global Manager is configured to filter out the component (raw trap) events. If desired, the Global Console operator can display the component events from the Notification Properties window by clicking the Aggregate tab.

Advanced SNMP trap integration

For more sophisticated SNMP trap processing, you can define an ASL script in a trap definition that is invoked during the processing of the trap. To define an ASL script, you add the keyword ASL, followed by the ASL script name, to the trap definition. For example:

```
BEGIN_TRAP  .1.3.6.1.4.1.9.10.1  0  0
 ClassName:   System
 ASL:         my-trap-rules.asl
 InstanceName:$SYS$
 EventName:   ColdStart
 Type:        MOMENTARY
END_TRAP
```

Notification attribute values are initially determined by the trap definitions. If an attribute value has not been defined by a trap definition, then the value for the attribute will come from the default section. The notification attribute values are exported from within the ASL script. The ASL script can override default attribute values. After the ASL script completes, the notification is created with those sets of values.

The ASL script relies on two special types of variables: input/output and input-only. Input/output variables correspond to attributes of the notification and the parameters in the configuration file. The input-only variables correspond to the variables used in the configuration file (for example, SYS, $V4$, and N).

A DISCARD_TRAP variable is included with the input/output variables. The variable controls whether a given trap message is discarded or processed. The default value for this variable is “NO,” which means the trap must be processed. To override the default and discard the trap, create a rule in the functional loop of the hook script in my_trap_rules.asl to process the trap, and within this rule set the DISCARD_TRAP to “YES.”

To run the ASL script, you must place it in the BASEDIR/smarts/local/rules/icoi-trapd folder. Further, ensure that you include the hook script in the trap definition in the trap_mgr.conf file for the hook script to be invoked for a particular OID.

Example of ASL script to set EventText

In the following example, ASL script extracts the substring “Agent Restarted for Reason:” from the contents of a trap’s first varbind, and places the remaining string into the notification attribute EventText. In the script, a list of all of the input and output variables as well as the input-only variables precedes the start of the processing (START).

Only the variables used by the script need to be declared:

```
/*
 * The first variable binding is a string of the form:
 *  "Agent Restarted for Reason: <text to extract>"
 */
// Input/Output variables.
CLASSNAME = "";
INSTANCENAME = "";
EVENTNAME = "";
AGGREGATE = "";
SEVERITY = "";
EVENTTEXT = "";
CATEGORY = "";
COMMUNITY = "";
DISCARD = "";
FORCEOCCURREDON = "";
UPDATEUD = "";
EXPIRATION = "";
STATE = "";
INMAINTENANCE = "";
CLEARONACKNOWLEDGE = "";
TRAPSOURCE = "";
EVENTTYPE = "";
MAP = "";
ASL = "";
ELEMENTCLASSNAME ="";
ELEMENTNAME ="";
SYSNAMEORADDR ="";
UNKNOWNAGENT ="";
```

```
LOGFILE ="";
USERDEFINED1 = "";
USERDEFINED2 = "";
USERDEFINED3 = "";
USERDEFINED4 = "";
USERDEFINED5 = "";
USERDEFINED6 = "";
USERDEFINED7 = "";
USERDEFINED8 = "";
USERDEFINED9 = "";
USERDEFINED10 = "";
DISCARD_TRAP = "YES";
// Input Variables
TIMESTAMP = "0";
IPADDRESS = "";
ENTERPRISE = "";
GENERIC = "9999";
SPECIFIC = "9999";
V1 = "";
V2 = "";
V3 = "";
V4 = "";
V5 = "";
V6 = "";
V7 = "";
V8 = "";
V9 = "";
V10 = "";
V11 = "";
V12 = "";
V13 = "";
V14 = "";
V15 = "";
V16 = "";
V17 = "";
V18 = "";
V19 = "";
V20 = "";
START {
 PARSE_EVENT_TEXT
}
PARSE_EVENT_TEXT {
 input = V1;
 ..  "Reason:"  EVENTTEXT: rep(word)  eol
}
```

The processing of the trap's first varbind occurs in the rule PARSE_EVENT_TEXT. The input to the rule is the contents of the first varbind. The line after the input reads all text up to and including "Reason:". The rest of the string is assigned to the variable EVENTTEXT. Any change to the value of the input/output variables automatically gets placed in the corresponding attribute in the translated Service Assurance notification. ASL Reference Guide provides more information about ASL.

Verifying the SNMP Trap Adapter port setting

The default value for the SNMP Trap Adapter (trap receiver) port is 9000. The System Administration Guide provides more information about how to change this default value, and about how to modify runtime parameters for an application.

SNMP trap batching

By default, the SNMP Trap Adapter immediately converts traps it receives (that match a trap definition) into new Service Assurance notifications or into updates to existing Service Assurance notifications and forwards them on to the Global Manager. SNMP trap batching refers to a process where the Adapter Platform Server waits for a specified period of time before forwarding re-notifications (updated notifications) to the Global Manager.

In case of high frequency of traps, you can use batching to improve performance of clients processing the converted Service Assurance notifications. You configure batching by editing your local copy of trap_mgr.conf file so that traps renotifying an event are held for a specified period of time. After that time is exceeded, only the most recent trap of those bearing the same notification name is sent to the Global Manager.

To set the batch parameter, use **sm_edit** to open *BASEDIR*/smarts/local/conf/icoi/trap_mgr.conf, and type the period of time in seconds you want the trap receiver to wait before forwarding re-notifications to the Global Manager. The following text shows the section where you set the batch parameter. By default, the batch setting is 10 seconds. To disable batching, specify zero (0):

```
# This interval (in seconds) will be used to batch updates to # notifications. In case, where
a high frequency of
# notifications occur, batching will improve performance.
# Setting this interval to 0 will disable batching.
 BATCH_NOTIFY_INTERVAL = 10
```

Using keywords in trap_mgr.conf

Keywords in trap_mgr.conf lists and describes keywords that you can set in your local trap_mgr.conf.

Table 3-7. Keywords in trap_mgr.conf

Keyword	Description	Option or format
BATCH_NOTIFY_INTERVAL	Allows you to set an interval (in seconds) to batch updates to notifications. In case, where a high frequency of notifications occurs, batching improves performance. To disable batching, set this interval to 0. For example: BATCH_NOTIFY_INTERVAL = 10	Any integer
CACHE_HOSTS	**Note** Use this CACHE_HOSTS keyword only if the *"RESOLVE_IP"* keyword is TRUE. Allows you to determine whether the system hosts will be cached and used to resolve IP addresses instead of calling the system functions, which may be slow in some cases. **Note** This is ONLY useful if DNS name resolution is disabled in the environment. This is sometimes done to improve performance, but it requires an extensive hosts table. By altering the *"HOSTS_PATH"* keyword, you can use a special file to load the hosts cache instead of using the same hosts file used by the system that the trap processor is running on. For example: CACHE_HOSTS = FALSE	TRUE or FALSE
DEBUG	Allows you to generate status messages, which you can use to debug the trap processor —without updating the trap_mgr_parse.asl. When DEBUG is set to TRUE in your local trap_mgr.conf, verbose status messages are generated, and your trap processor runs slower while the log grows faster. For example: DEBUG = FALSE	TRUE or FALSE
HOSTS_PATH	**Note** Use this HOSTS_PATH keyword only if *"CACHE_HOSTS"* is TRUE. Allows you to define the full path to the system hosts file. (You can use a unique file for the cache by specifying its path here.) For example: HOSTS_PATH = /etc/hosts	Path name

Table 3-7. Keywords in trap_mgr.conf (continued)

Keyword	Description	Option or format
IMPORT	Allows you to access external information or processes for processing traps—without editing ASL scripts. You can use the IMPORT keyword in your local trap_mgr.conf to load tables, or any other pre-processor driver scripts. To load import drivers and lookup tables, you can replicate the IMPORT keyword as many times as necessary. For example: IMPORT TRUE TRUE driver1 import_rules.asl local/conf/someinput.txt |	The format is as follows: IMPORT <wait> <auto-reload> <driver-name> <rules-file> [<input-file>] ... [<field-separator>] where: ■ <*wait*> is TRUE or FALSE. Set the Trap Processor to wait for the driver to exit. ■ <*auto-reload*> is TRUE or FALSE. Set the Trap Processor to start this driver using a **--reloadConfig** call. ■ <*driver-name*> is simple string used to name the driver. ■ <*rules-file*> is the name of the rules file located in rules/icoi-rules. ■ <*input-file*> is optional. Set the path to an input file from $SM_SITEMOD. ■ <*field-separator*> is optional. Set the field separator character.
LOGGING	Allows you to generate status messages, which you can use to capture information about the trap processor. When LOGGING is set to OFF, for example, in your local trap_mgr.conf, error messages are generated and sent to the log file. For example: LOGGING = OFF	ALL, DISCARD, STATUS, and OFF: ■ ALL logs all messages about traps received and discarded, as well as server status. ■ DISCARD logs messages about discarded traps and server status. ■ STATUS logs messages about changes to the server status. ■ OFF logs error messages.

Table 3-7. Keywords in trap_mgr.conf (continued)

Keyword	Description	Option or format
LOOKUP	Allows you to substitute one value for another in the trap processor—without editing ASL scripts. You can use the LOOKUP keyword in your local trap_mgr.conf to substitute values. For example: LOOKUP MyTable local/lookups/filename In this example, filename contains: abcd1234:Alexandria stuvwxyz:Hopkinton If the trap definition in your trap_mgr.conf includes: UserDefined1: $MyTable($SYS$)$ Then, if SYS contains abcd1234, then UserDefined1 returns the value Alexandria. If SYS contains stuvwxyz, then UserDefined1 returns the value Hopkinton. Other examples include: $TABLE1($V4$)$; $TABLE2($SYS$[4])$ $TABLE2(coldStart[1])$	Available formats include: ■ <*table*>(<*key*>) accesses a lookup table to return a single element if the table has only one value, or a list if the table has more than one value. ■ <*table*>(<*key*>[<*index*>]) accesses a lookup table to return an element from the table. where: ■ <*table*> is the name of the table defined in the IMPORT keyword. ■ <*key*> is the key into this table, which is any string or variable. ■ <*index*> is an optional index value, which starts from 1 if the table contains more than one value.

Table 3-7. Keywords in trap_mgr.conf (continued)

Keyword	Description	Option or format
RESOLVE_IP	Allows you to resolve IP addresses into their names. You can use this option to determine whether unrecognized IP addresses will be resolved before being assigned to the SYS variable. For objects already known to the system, this option has no effect. **Note** Name resolution can be very slow. A system receiving thousands of traps and calling a name server for every request can seriously impact the number of traps that can be processed. Threading can help somewhat, but name resolution is *not recommended* in large installations unless the server is configured to use only host file resolution and the *"CACHE_HOSTS"* keyword is defined. Alternatively, a caching DNS server can be run on the trap host to help somewhat, but performance will still be impacted. For example: RESOLVE_IP = FALSE	TRUE or FALSE
USE_CURRENT_TIME	Allows you to set the timestamp for notifications—using either the time contained in the trap, or the time the trap was received. If this keyword is set to FALSE, then the timestamp from the trap is used unless it is invalid. If this option is set to TRUE, the timestamp from the trap is discarded, and the current time is used when creating notifications. **Note** When using threads, traps can appear to come in out of order if this keyword is enabled. CLEAR traps are reordered, but other traps appear to occur at the time they were processed. For example: USE_CURRENT_TIME = FALSE	TRUE or FALSE

sm_trapd options

Options for sm_trapd lists and describes available options for sm_trapd.

Table 3-8. Options for sm_trapd

Option	Description
--ascii	If “--ascii” is not specified on the commandline for starting sm_trapd, sm_trapd converts the entire value of that OID into a printable UTF-8 coding string, such as a varbind. For example, .1.3.6.1.4.1.333.1 -> "abcd012" is converted into .1.3.6.1.4.1.333.1 -> "61 62 63 64 C3 96 30 31 32". If “--ascii” is specified on the commandline for starting sm_trapd, sm_trapd converts only the non-printable characters into a HEX string using the UTF coding for these characters. The remaining printable characters of the original value (octet-string) remain unchanged, such as the varbind. For example, .1.3.6.1.4.1.333.1 -> "abcd012" is then converted into .1.3.6.1.4.1.333.1 -> "abcdXC3X96012."
--broker	Alternate broker location as host:port. For example: **--broker** AM_HOST-NAME:AM-PORT. Also -b <*location*>.
--community	Stream the trap community string to the parser. Writes the community string to the sm_trapd.log file. **Note** This option is required in the sm_trapd command line when using a community string in a hookscript or the C variable.
--config=<*cfg*>	Name of trap configuration directory where the trapd.conf is located. The trapd.conf file is loaded from one of the following directories: ■ *BASEDIR*/smarts/local/conf/<*cfg*> or ■ *BASEDIR*/smarts/conf/<*cfg*> The file in the local directory takes precedence if the configuration is present in both places. **Note** Default is **--config**=trapd.
--limitQueueMegs=<*num*>	Size limit of internal trap queue. When the limit is reached, discard incoming traps from the most active sources.
--model=<*model*>	Name of the model library that you want to load. For example: -model=sm_actions. Also -m <*model*>.
--name=<*name*>	Registered server name you want to start. Also -name <*name*>.
--output	Data transfer.
--port=<*port*>	Alternate port number for receiving traps. Also -p <*port*>. **Note** Default value is **--port**=162.
--reloadConfig	Reload, rather than restart, the trap configuration file for an already-running SNMP trap adapter. For example: -s <*adaptername*> **--reloadConfig**.
--rules=<*name*>	Name of the rules set for parsing traps. For example: **--rules**=icoi-trapd/trap_mgr_parse.asl.
--rules=default	Name of the default rules set for parsing traps. **Note** Default is **--rules**=*BASEDIR*/smarts/rules/trapd/trapParse.asl.
--seed=<*file*>	Get SNMPv3/USM credentials from the seed file. For example: **--seed**=seedfile. The seed file has a path that is relative to the conf directory.
--server=<*name*>	Name of the server. For example: **--server**=INCHARGE-OI. Also -s <*name*>.

Table 3-8. Options for sm_trapd (continued)

Option	Description
--source	Forward the source address of the trap to the trap processor and make this address available to configuration files and ASL scripts.
--sport=<*port*>	Alternate registration port. Use with **--name**. For example: **--sport**=9180. Also -P <*port*>.
--tag	Tag each varbind value with its type.
--window=<*num*>	Window size for duplication. Also -w <*num*>.
Trace options	
--traceSNMP	Trace incoming SNMP messages.
--traceRules	Trace rule compilation.
--traceServer	Trace interactions with the back-end server.
--traceParse	Trace rule matching.
Standard options	
--accept=<*host-list*>	Accept connections only from hosts on <*host-list*>, which is a comma-separated list of hostnames and IP addresses. The any option for <*host-list*> allows any host to connect. **Note** Default is **--accept**=any.
--daemon	Run process as a daemon.
--errlevel=<l*evel*>	Minimum error printing level. **Note** Default is **--errlevel**=Warning.
--help	Print help and exit.
--loglevel=<*level*>	Minimum system logging level. **Note** Default is **--loglevel**=Error.
--logname=<*name*>	Use <*name*> to identify sender in the system log. **Note** Default is the name of a program.
--output[=<*file*>	Redirect server output (stdout and stderr). The filename is <*file*>, or the **--logname** value if <f*ile*> is omitted. Log files are in $SM_LOGFILES or $SM_WRITEABLE/logs.
--tracelevel=<l*evel*>	Minimum stack trace level. Options for <*level*> include: One of None, Emergency, Alert, Critical, Error, Warning, Notice, Informational, or Debug. Fatal is a synonym for Critical. For example: **--tracelevel**=Informational. **Note** Default is **--tracelevel**=Fatal.
--version	Print program version and exit.
--useif=<*ip-address*>	Use this IP address as the source or destination interface address.

Table 3-8. Options for sm_trapd (continued)

Option	Description
--	Stop scanning for options.
--trailingNulls	This option allows for printing trailing NULL bytes in an octet string of varbind value as 00 or \X00. By default trailing null octets are ignored. For example: Device data: 41 00 00 00 00 sm_adapter data: 41 If sm_adapter is started with option --trailingNulls, then the adapter will print the octet string as it is, that is, the way it is coming from the trap device, including the trailing null octets. For example: sm_adapter started with --trailingNulls option: Device data: 41 00 00 00 00 sm_adapter data: 41 00 00 00 00

Starting and stopping the SNMP Trap Adapter

If you installed the SNMP Trap Adapter as a service, it automatically starts when the system starts up. The following instructions describe how to use the **sm_service** utility to manually start and stop the SNMP Trap Adapter.

Note To use the **sm_service** utility to install a service or start a service, you must have administrative privileges on the local host.

Issue one of the following from the command line:

```
BASEDIR/smarts/bin/sm_service start ic-trapd-receiver
```

or

```
BASEDIR/smarts/bin/sm_service stop ic-trapd-receiver
```

The System Administration Guide provides more information about the **sm_service** utility and about how to modify **sm_service** options.

Default sm_service parameters for the SNMP Trap Adapter

When the SNMP Trap Adapter is installed as a service during the installation process, the SNMP Trap Adapter is set up with default parameters. The default parameters and their values are:

```
sm_service install --force  --unmanaged  --startmode=runonce \
'--name=ic-trapd-receiver' \
'--description=VMware Smart Assurance SNMP Trap Adapter' \
'/opt/InCharge/SAM/smarts/bin/sm_trapd' \
```

```
'--name=TRAP-INCHARGE-OI' \
'--server=INCHARGE-OI' \
'--config=icoi' \
'--port=162' \
'--model=sm_actions' \
'--rules=icoi-trapd/trap_mgr_parse.asl' \
'--seed=seedfile' \
'--output=TRAP-INCHARGE-OI.log'
```

Syslog Adapter

4

Read the following topics next:

- Overview
- Configuring the Syslog Adapter
- Starting and stopping the Syslog Adapter

Overview

The Syslog Adapter tails and parses the contents of any system log file and generates notifications to the Global Manager based on the file contents.

You start the adapter by invoking the **sm_service** command. For example:

```
InCharge/SAM/smarts/bin>sm_service start ic-syslog-adapter
```

First the syslog_mgr.asl file parses each syslog message and populates the input variables. Then the my_hook_syslog.asl gets executed. This script first populates the notification attribute output variables by using the default values. It then uses the MODIFY_ATTRIBUTES rule to set additional attributes defined and potentially modify (overriding) any default attribute values. This is how the Service Assurance notification gets created by the Syslog Adapter.

Before you configure the Syslog Adapter, identify the location of the SYSFILE you want the adapter to tail and parse and ensure that **sm_service** install command line for the ic-syslog-adapter identifies this location. You must also ensure that the file format of the Syslog exactly matches the format described in the following sections.

Configuring the Syslog Adapter

Syslog Adapter configuration and script files describes the files that you need to use when configuring the Syslog Adapter. If you edit any of these files, you must use the **sm_edit** utility. The utility will save the local copies to the appropriate subdirectories under the *BASEDIR*/smarts/local directory. The sm_edit utility provides more information.

Table 4-1. Syslog Adapter configuration and script files

Directory under *BASEDIR*/	Filename	User editable	Description
smarts/rules/icoi-syslog/	my_hook_syslog.asl	Yes	Basic template for processing a syslog file.
smarts/rules/icoi-syslog/	syslog_mgr.asl	Yes	Rule set for parsing each syslog message.

The Syslog Adapter creates events by parsing the contents of syslog files. You can use it to parse the contents of any text file with entries of the format:

```
month day time hostName applicationName [process_id]:text_message
```

If the format of your syslog file is different from the above format, you can edit my_hook_syslog.asl and syslog_mgr.asl to parse the entries accordingly.

The Syslog Adapter can parse the contents of a file and it can tail a file. When the Syslog Adapter tails a file, it skips the existing content and uses only content added to the file while the adapter is running.

Note The process_id parameter is optional when parsing the contents of syslog files.

The Adapter Platform includes a basic template for processing a syslog file. This file is *BASEDIR*/smarts/rules/icoi-syslog/my_hook_syslog.asl.

After ensuring that the Adapter Platform Server and the Global Manager are up and running, complete the following procedures to configure the Syslog Adapter:

1 Check the location of the Syslog file to be sure it is appropriately placed for your operating system.

2 Change the parameters in the local copy of my_hook_syslog.asl to match your needs.

3 Start the Syslog Adapter.

Syslog file location

The **sm_service** install command line for the Syslog Adapter, ic-syslog-adapter, is stored in the **sm_service** database. By default, the SYSFILE parameter in this script specifies the location as /var/log/syslog. Edit the command line to change the location of the Syslog file if necessary.

Editing the my_hook_syslog.asl

You can use the Adapter Scripting Language (ASL) to modify the functionality of the local copy of my_hook_syslog.asl or to create a new file. The basic components of a custom processing file for the Syslog Adapter are explained below:

```
debug = FALSE;
ASLNAME = " ".getRuleFileName ().": ";
DISCARD = "TRUE";
CLEAR_SYSLOG = "FALSE";
```

```
BATCH_NOTIFY_INTERVAL = 10;
// Output variables : This section has all default settings.
CLASSNAME = "Syslog";
INSTANCENAME = "";
EVENTNAME = "";
SEVERITY = "2";
EVENTTEXT = "";
CATEGORY = "";
EXPIRATION = "7200";
STATE = "";
INMAINTENANCE = "FALSE";
CLEARONACKNOWLEDGE = "TRUE";
EVENTTYPE = "";
USERDEFINED1 = "";
USERDEFINED2 = "";
USERDEFINED3 = "";
USERDEFINED4 = "";
USERDEFINED5 = "";
USERDEFINED6 = "";
USERDEFINED7 = "";
USERDEFINED8 = "";
USERDEFINED9 = "";
USERDEFINED10 = "";
ELEMENTCLASSNAME = "";
ELEMENTNAME = "";
SYSNAMEORADDR = "";
UNKNOWNAGENT = "IGNORE";
LOGFILE = "NONE";
// Aggregate Section :
AGG_EVENTNAME = "";
AGG_ELEMENTNAME = "";
AGG_EVENTTEXT = "";
```

The values of the output variables populate the attributes of the standard notification created when the syslog message is imported. The variable names correspond directly to the standard notification's attribute names.

The Syslog Adapter populates these variables when the syslog entry is parsed:

```
// Input Variables
SYSLOGTIME = "";
HOST = "";
APPLICATION_NAME = "";
PROCESS_ID = "";
MESSAGE = "";
```

The START rule takes the text parsed from the syslog entry as input, prints a message, calls three other rules, prints another message, and exits after the processing is complete:

```
START {
  input=MESSAGE;
do {
if (debug) {print(time().ASLNAME."SYSLOGTIME =".SYSLOGTIME);}
if (debug) {print(time().ASLNAME."HOST =".HOST);}
```

```
if (debug) {print(time().ASLNAME."APPLICATION_NAME=".APPLICATION_NAME);}
if (debug) {print(time().ASLNAME."PROCESS_ID =".PROCESS_ID);}
if (debug) {print(time().ASLNAME."MESSAGE =".MESSAGE);}
}
    PARSE_MESSAGE
    MODIFY_ATTRIBUTES
    CUSTOM_RULE?
} do {
    if (debug) { print(time().ASLNAME."Done with my_hook_syslog.asl ");}
    return;
}
```

The CUSTOM rule is an example of a rule which performs more customizations. In this case, it saves a prefix and a message description:

```
CUSTOM_RULE {
    unusedPrefix:rep(notany(":")) ":" /* consume chars up to : */
    msgDescription:rep(word) eol
} do {
    if (debug) { print(time().ASLNAME."Executing CUSTOM_RULE");}
}
```

This PARSE_MESSAGE rule saves only the first 30 characters:

```
PARSE_MESSAGE {
} do {
    // Use a slice of 30 characters as part of EVENTNAME
    slice = substring(MESSAGE, 0, 30);
}
```

The MODIFY_ATTRIBUTES rule assigns values to the notification created from the syslog entry. The value of InstanceName is composed of HOST, APPLICATION_NAME, and PROCESS_ID. These are values parsed from the syslog entry:

```
/*
 * MODIFY_ATTRIBUTES Rule:
 * All your customizations are done here. You can use all
 * the Syslog input variables wherever you want them assigned
 * to ICS_Notification attributes.
 * ------------------------------------------------------------ */
MODIFY_ATTRIBUTES {
} do {
  DISCARD = "TRUE";
  CLEAR_SYSLOG = "FALSE";
  BATCH_NOTIFY_INTERVAL = 10;
  CLASSNAME = "Syslog" ? LOG;
  INSTANCENAME = HOST."_".APPLICATION_NAME."_".PROCESS_ID ? LOG;
  EVENTNAME = slice ? LOG;
   SEVERITY = "2" ? LOG;
   EVENTTEXT = MESSAGE ? LOG;
   CATEGORY = "" ? LOG;
   EXPIRATION = "7200" ? LOG;
   STATE = "NOTIFY" ? LOG;
   INMAINTENANCE = "FALSE" ? LOG;
```

```
    CLEARONACKNOWLEDGE = "TRUE" ? LOG;
    EVENTTYPE = "DURABLE" ? LOG;
    USERDEFINED1 = "" ? LOG;
    USERDEFINED2 = "" ? LOG;
    USERDEFINED3 = "" ? LOG;
USERDEFINED4 = "" ? LOG;
USERDEFINED5 = "" ? LOG;
USERDEFINED6 = "" ? LOG;
USERDEFINED7 = "" ? LOG;
USERDEFINED8 = "" ? LOG;
USERDEFINED9 = "" ? LOG;
USERDEFINED10 = "" ? LOG;
ELEMENTCLASSNAME = "Host";
ELEMENTNAME = HOST;
SYSNAMEORADDR = HOST;
UNKNOWNAGENT = "CREATE";
LOGFILE = "NONE";
AGG_EVENTNAME = "AggEvent-".INSTANCENAME;
AGG_ELEMENTNAME = HOST;
AGG_EVENTTEXT = "This is an Aggregate Test"
}
```

You can optionally add logic that compares the input variables and sets the output variables based on them.

Whenever you modify the hook script file, you must restart the adapter for the changes to take effect.

Syslog Adapter parameters

Syslog Adapter parameters shows the complete list of parameters that can be defined for a notification forwarded to the Global Manager by the Syslog Adapter.

Note If no parameter values are set within the MODIFY_ATTRIBUTES rule, then the values set in the Output variables section are used.

Table 4-2. Syslog Adapter parameters

Parameter	Corresponding standard notification attribute or description	Valid values
AGG_EVENTNAME	This is the event name of the aggregate notification and is required if you are creating an aggregate.	<*string*>
AGG_ELEMENTNAME	This is the name of the topology element where the aggregate event occurs. This field is optional. If this field is not defined, the ElementClassName and ElementName of the trap definition are used, if available. Otherwise, the ClassName and InstanceName are used. If you define this field and specify an object-handle that does not exist in the topology, then the aggregate is not created.	<*object-handle*>
AGG_EVENTTEXT	This is the description of the aggregate notification (event). This field is optional.	<*string*>
CATEGORY	Category	Any string

Table 4-2. Syslog Adapter parameters (continued)

Parameter	Corresponding standard notification attribute or description	Valid values
CLASSNAME	ClassName and DisplayClassName	Any string
CLEARONACKNOWLEDGE	ClearOnAcknowledge	TRUE or FALSE
CLEAR_SYSLOG	Controls whether a given syslog message clears an already existing notification or results in a notify. The default is FALSE, all syslog messages result in notifications. All syslog messages result in notifications unless the syslog messages are processed by subsequent MODIFY_ATTRIBUTES rules, which override the default.	TRUE or FALSE
DISCARD	Controls whether a given syslog message is discarded or processed. The default is TRUE, all syslog messages are discarded. To override the default and process a syslog message, create a rule to process this syslog message. Within this rule, DISCARD should be set to FALSE.	TRUE or FALSE
ELEMENTCLASSNAME	ElementClassName	Valid Class Name in topology
ELEMENTINSTANCENAME	ElementName	Valid Instance Name in topology
EVENTNAME	EventName (describing the notification) and the EventDisplayName	Any string
EVENTTEXT	EventText	Any string
EVENTTYPE	EventType	MOMENTARY or DURABLE
EXPIRATION	Defines the expiration time in seconds for the notification. Zero indicates that the notification will not expire. 7200 is the default value. If you use zero, you should use some method to eventually clear the notification. For example, you can configure the Syslog Adapter to send clear state notifications or set the ClearOnAcknowlege parameter to TRUE.	Integer
INMAINTENANCE	InMaintenance	TRUE or FALSE
INSTANCENAME	InstanceName (of the object associated with the notification) and InstanceDisplayName	Any string
LOGFILE	The name of the file used by the SNMP Trap Receiver to log information for this trap. If this parameter is 'NONE' or not defined, no information is logged.	Filename
SEVERITY	Severity	Any integer from 1 to 5
STATE	Describes the state of the notification. This option is deprecated; its function is replaced by CLEAR_SYSLOG.	NOTIFY or CLEAR

Table 4-2. Syslog Adapter parameters (continued)

Parameter	Corresponding standard notification attribute or description	Valid values
SYSNAMEORADDR	A valid value describes either the system name or address of the entity that sent the trap. Values for this parameter override values in ElementClass and ElementInstance.	Any string
UNKNOWNAGENT	Describes whether Adapter Platform should ignore traps related to unknown topology elements or create elements for the traps.	CREATE or IGNORE
USERDEFINED1-10	UserDefined1-10	Any string

Using the Syslog Aggregate parameters

You configure aggregates in the Aggregate Section of the local copy of my_hook_syslog.asl rule set, located in *BASEDIR*/smarts/local/rules/icoi-syslog directory. Using the Trap Adapter Aggregate parameter provides general information about Aggregates.

The following example illustrates how to use the aggregate parameter with the Syslog Adapter:

```
/*
 * my_hook_syslog.asl - Hook adapter for
 * any syslog related customizations.
 *
 * Copyright (C) 1997, System Management ARTS (SMARTS)
 * All Rights Reserved
 */
debug = FALSE;
ASLNAME = " ".getRuleFileName().": ";
DISCARD = "TRUE";
CLEAR_SYSLOG = "FALSE";
/*
 * This interval (in seconds) will be used to batch updates to
 * notifications. In case, where a high frequency of
 * notifications occur, batching will improve performance.
 * Setting this interval to 0, will disable batching.
 */
BATCH_NOTIFY_INTERVAL = 10;
CLASSNAME = "Syslog";
INSTANCENAME = "";
EVENTNAME = "";
SEVERITY = "2";
EVENTTEXT = "";
CATEGORY = "";
EXPIRATION = "300";
STATE = "";
INMAINTENANCE = "FALSE";
CLEARONACKNOWLEDGE = "TRUE";
EVENTTYPE = "";
USERDEFINED1 = "";
USERDEFINED2 = "";
USERDEFINED3 = "";
USERDEFINED4 = "";
```

```
USERDEFINED5 = "";
USERDEFINED6 = "";
USERDEFINED7 = "";
USERDEFINED8 = "";
USERDEFINED9 = "";
USERDEFINED10 = "";
ELEMENTCLASSNAME = "";
ELEMENTNAME = "";
SYSNAMEORADDR = "";
UNKNOWNAGENT = "IGNORE";
LOGFILE = "NONE";
/* Need to Declare these, if you want Aggregates
 * ------------------------------------------- */
AGG_EVENTNAME = "";
AGG_ELEMENTNAME = "";
AGG_EVENTTEXT = "";
/*
 * Input Variables: Following are the variable declarations,
 * which hold the Syslog parsed values.
 * ---------------------------------------------------------- */
SYSLOGTIME = "";
HOST = "";
APPLICATION_NAME = "";
PROCESS_ID = "";
MESSAGE = "";
if (debug) {  print(time().ASLNAME."Activated"); }
/*
 * Start Rule
 * ---------- */
START {
    input=MESSAGE;
    MODIFY_ATTRIBUTES
    CREATE_AGGREGATE
} do {
    if (debug) { print(time().ASLNAME."Done with my_hook_syslog.asl ");}
    return;
}
CREATE_AGGREGATE {
} do {
        // If you see strings "CPU" and "HighUtilization" in
        // the syslog
        // message, then generate and aggregate.
        // -------------------------------------------------
        if (glob("*CPU*",MESSAGE) &&
        glob("*HighUtilization*",MESSAGE)) {
                AGG_EVENTNAME = "Degraded";
                AGG_ELEMENTNAME = HOST;
                AGG_EVENTTEXT = "Host [".HOST."] is Degraded";
        }
}
MODIFY_ATTRIBUTES {
} do {
    CLASSNAME = "Processor" ? LOG;
    INSTANCENAME = "PRO-".HOST ? LOG;
    EVENTNAME = substring(MESSAGE, 0, 30) ? LOG;
```

```
    SEVERITY = "2" ? LOG;
    EVENTTEXT = MESSAGE ? LOG;
    CATEGORY = "" ? LOG;
    EXPIRATION = "7200" ? LOG; //PR:6617
    STATE = "NOTIFY" ? LOG;
    INMAINTENANCE = "FALSE" ? LOG;
    CLEARONACKNOWLEDGE = "TRUE" ? LOG;
    EVENTTYPE = "DURABLE" ? LOG;
    ELEMENTCLASSNAME = "Processor";
    ELEMENTNAME = "PRO-".HOST ? LOG;
    UNKNOWNAGENT = "CREATE";
    LOGFILE = "Processor.log";
}
DEFAULT {
     msg:{.. eol}
} do {
    print(time().ASLNAME."Reached Default rule: ".msg);
    this->clearVariables();
}
/*
 * These variables describe the formatting of this file.  If
 * you don't like the template defaults, feel free to change
 * them here (not in your .emacs file).
 *
 * Local Variables:
 * mode: C++
 * End:
 */
```

Syslog batching

By default, except if the DISCARD parameter is set to TRUE, the Syslog Adapter does the following:

- Converts the syslog messages it receives into new Service Assurance notifications or updates the existing Service Assurance notifications.
- Forwards them on to the Global Manager.

  Syslog batching refers to a process where the Adapter Platform Server waits for a specified period of time before forwarding re-notifications (updated notifications) to the Global Manager.

  In case of high frequency of syslog messages, you can use batching to improve performance of clients processing the converted Service Assurance notifications. You configure batching by editing the my_hook_syslog.asl file so that re-notification messages are held for a specified period of time. Then, once that time is exceeded, only the most recent message of those bearing the same notification name is sent to the Global Manager.

To set the batch parameter, use **sm_edit** to open my_hook_syslog.asl, and type the period of time (in seconds) you want the Syslog Adapter to wait before forwarding re-notifications to the Global Manager. By default, the batch setting is 10 seconds. To disable batching, specify zero (0):

```
BATCH_NOTIFY_INTERVAL = 10
```

Using the Syslog Aggregate parameters provides an example of a script that contains the batch parameter.

Starting and stopping the Syslog Adapter

If you installed the Syslog Adapter as a service, it automatically starts when the system starts up. The following instructions describe how to use the **sm_service** utility to manually start and stop the Syslog Adapter.

Note To use the **sm_service** utility to install a service or start a service, you must have administrative privileges on the local host.

Issue one of the following from the command line:

```
        BASEDIR/smarts/bin/sm_service start ic-syslog-adapter
```

or

```
        BASEDIR/smarts/bin/sm_service stop ic-syslog-adapter
```

The System Administration Guide provides more information about the **sm_service** utility and about how to modify **sm_service** options.

Default sm_service parameters for the Syslog Adapter

When the Syslog Adapter is installed as a service during the installation process, the Syslog Adapter is set up with default parameters.

The default parameters and their values are:

```
t /InCharge/SAM/smarts/bin/sm_service install
--startmode=runonce
--description="SMARTS Syslog Adapter" ic-syslog-adapter
 /InCharge/SAM/smarts/bin/sm_adapter
    --name=SYSLOG-INCHARGE-OI
    --rserver=INCHARGE-OI
    --tail=/var/log/syslog
    --model=sm_system
    --model=sm_actions
```

```
    --output
    icoi-syslog/syslog_mgr.asl s
```

Command Line Interface

5

The command line interface (**sm_ems**) is useful for converting information from third-party applications into events. This interface can create and clear events and create basic topology elements. The command line interface also can update event attributes passed by the Adapter PlatformServer, and associate events to topology elements. You can use the ASL scripting language in conjunction with the command line interface for more advanced processing.

Read the following topics next:

- Using the command line interface

Using the command line interface

Configuration of the command line interface consists of two parts:

1 You need to set up the proper security between the Adapter Platform and the command line interface. By default, the **sm_ems** command line interface does not prompt for authentication credentials.

   If you are working in a secure environment, you need to create an entry in the clientConnect.conf file (using the **sm_edit** utility) that provides the necessary username and password for the **sm_ems** interface. The username in this entry must match a user in the corresponding serverConnect.conf file. You must edit the clientConnect.conf file on the host where **sm_ems** is running.

2 You must modify your third-party application to call **sm_ems**.

Command line interface usage

The third-party application must be modified to call the **sm_ems** command. The documentation for third-party applications provides more information about configuring the application to call the command line interface.

The basic format to call the command line interface is:

```
sm_ems --server=<server_name> [options ...] <command>
```

Command line interface options describes the options.

Table 5-1. Command line interface options

Option	Description
--server=<*name*>	The name of the Adapter Platform Server. This parameter is required.
--broker=<*location*>	The name of the broker, if you override the broker set by the SM_BROKER environment variable.
--system=<*nameOrAddr*>	Specifies the name or IP address of the existing system (Unitary Computer System) you want to associate with this event. The event will automatically be associated with this system in the Adapter Platform topology. The system name is converted to its canonical name using hostname lookups. For example, Host1 or 10.1.2.345 might be converted to Host1.example.com. If the system does not exist in the topology, it may be created automatically if the **--create-system** option is specified.
--create-system	Determines that the unitary computer system should automatically be created if it does not exist in the topology. The Class defaults to Node, however you can optionally use **--element-class** to specify the class name. This option is deprecated, but retained for backwards compatibility. The **--create-element** option can be used instead of this option to create any infrastructure element.
--element-class=<*ClassName*>	Determines the ClassName of the entity being created in the topology, if the element does not already exist. Issue this option with **--create-element** and also **--element-name**.
--element-name=<*InstanceName*>	Determines the InstanceName to be used with the **--element-class** option. Options provided with **--element-class** and **--element-name** will be used to create the object. **Note** Do not use the **--system** option if you are also issuing the **--element-class** and **--element-name** options.
--create-element	Determines that the element-class and element-name you specify should automatically be created if it does not exist in the topology. Use this option to create any infrastructure element, including business elements, application-related elements, and unitary computer systems.
--aggregate-element-class=<*Class*Name>	Determines the ClassName for the Aggregate Notification, if one is being created.
--aggregate-element-name=<*Instan*ceName>	Determines the InstanceName for the Aggregate Notification, if an aggregate notification is being created (using the **--aggregate-event** option). Also creates an OccurredOn relationship between the Instance and the Aggregate Event.
--aggregate-event=<*AggregateEvent*tName>	Determines the EventName of an Aggregate Notification. This is a required option if you are creating an Aggregate.
--audit=<*msg*>	Optional text to include in the description field of the audit log entry created for this action. **Note** This option is ignored for the add-audit-log command.
--traceServer	Enables tracing of server communications.

Command line interface commands

There are 10 different commands to use with **sm_ems**. Along with the command, you need to pass arguments. sm_ems commands describes these commands. Where possible in the table, the arguments passed with the command are expressed as attributes from the standard notification. Notification attributes are described in Chapter 6 Adapter Platformand Notifications.

With the exception of summarize, all of the **sm_ems** commands use ClassName, InstanceName, and EventName. These three fields are always required because they uniquely identify a standard notification.

Table 5-2. sm_ems commands

Command	Description
acknowledge <*ClassName*> <*InstanceName*> <*EventName*>	Changes the notification's attribute, Acknowledged to TRUE. Attribute values modified using this command argument do not get propagated to Global Manager through the Adapter Platform.
add-audit-log <*ClassName*> <*InstanceName*> <*EventName*> <*message*>	Adds an entry to the Adapter Platform's audit log. The entry contains information about the notification as well as a text message (<*message*>).
assign <*ClassName*> <*InstanceName*> <*EventName*> <*Owner*>	Changes the notification's ownership. Attribute values modified using this command argument do not get propagated to Global Manager through the Adapter Platform.
clear <*ClassName*> <*InstanceName*> <*EventName*> <*SourceDomainName*>	Clears an occurrence of a notification. In order for this command to work, SourceDomainName must match the notification's attribute.
notify <*ClassName*> <*InstanceName*> <*EventName*> <*SourceDomainName*> <*EventType*> <*Clear-Mode*> [<*attribute*>=<*value*> ...]	Creates a notification. If you use the notify command to create a notification that already exists, the Count attribute is increased by 1 and any attribute which is not explicitly assigned a new value will retain its previous value. You must use values for each argument through <*Clear-Mode*>. Clear-Mode indicates how the notification gets cleared. Valid values for Clear-Mode are: ■ source — The source sends a clear. ■ <*number*> — The number of seconds before the event expires. ■ none — The notification should not expire and the source will not send a clear. At the end of this command, you have the option of defining other attributes for the command. Optional attributes are: Category, Certainty, ClassDisplayName, ClearOnAcknowledge, EventDisplayName, EventText, Impact, InMaintenance, InstanceDisplayName, Severity, TroubleTicketID, and UserDefined1 through UserDefined10. In this case, the attribute name must match the standard notification attribute. If you do not explicitly assign values for other attributes, the default attribute values are used. Chapter 6 Adapter Platformand Notifications provides a list of default values.
print <*ClassName*> <*InstanceName*> <*EventName*>	Prints all of the notification's attributes to stdout.
release <*ClassName*> <*InstanceName*> <*EventName*>	Clears the notification ownership. Attribute values modified using this command argument do not get propagated to Global Manager through the Adapter Platform.

Table 5-2. sm_ems commands (continued)

Command	Description
summarize [<*Notification_List*>]	Prints a summary for a given notification list. If a notification list is not specified, the notification list named Default is used.
unacknowledge <*ClassName*> <*InstanceName*> <*EventName*>	Changes the notification's attribute, Acknowledged to FALSE. Attribute values modified using this command argument do not get propagated to Global Manager through the Adapter Platform.
update <*ClassName*> <*InstanceName*> <*EventName*> <*attribute*>=<*value*> ...	Modifies one or more of a notification's attributes.

An example of the sm_ems Notify command

The following **sm_ems** command creates a notification on the Adapter Platform Server ICOI:

```
sm_ems --server=ICOI notify Router RouterNY25 Down
 3rdParty DURABLE 3600 Category=3rdParty-Down Severity=2 s
```

The notification is for a Router named RouterNY25 and it is Down, which is in this case a durable event. The notification expires in 3600 seconds. The Category attribute of the standard notification contains the name of the event as seen in the third-party application.

An example of the sm_ems Create Element option with Notify command

The following **sm_ems** command uses the **--create-element** option along with the notify command to:

- Create a Host, WebHost (Host::WebHost) if it does not exist in the topology.
- Generate a notification, ApplicationService::WebServer Stopped. Specifies that the event was sent by a third party, is a momentary type of event, will be cleared by its source, and has a severity level of 2. Also, implies that the event OccurredOn Host::WebHost:

```
sm_ems --server=ICOI --element-class=Host
--element-name=WebHost --create-element notify ApplicationService WebServer Stopped
3rdParty momentary
source severity=2 s
```

An example of the sm_ems Aggregate option with Notify command

The following **sm_ems** command uses **Aggregate** option along with the notify command to generate two notifications:

- Generate an Aggregate notification, CPU::WebHost-CPU High-Utilization. Specifies that the event was sent by a third party, is a momentary type of event, and has a severity level of 1.

- Generate a component Notification, Host::WebHost Unresponsive, which is associated with the previously generated Aggregate Notification:

```
sm_ems --server=ICOI --aggregate-element-class=Host --aggregate-element-name=WebHost --
aggregate-event-name=unResponsive notify CPU WebHost-CPU High-Utilization 3rdParty
momentary source severity=1 s
```

Adapter Platformand Notifications

6

The purpose of the Service Assurance Adapter Platformis to normalize events and place them into a topological context so that they can be imported into a Global Manager. This means that the Adapter PlatformServer takes incoming events from a variety of other sources and translates the information into standard notifications. The standard notification is based on the ICIM_Notification class and consists of many attributes. When the Adapter Platformis used as a platform for ASM Adapters, the ASM Adapters set the properties of the notification attributes. Depending on the configuration of the adapter, the attributes created for each notification may or may not be populated with data.

Also, when the Global Managercollects the notification information from the Adapter PlatformServer, not all of the attributes of the notification get passed by default. Notification attributeslists the attributes of the notifications for Service Assuranceand also identifies those that get passed to the Global Manager.

Note For attributes that contain a time value, time is counted in seconds from Midnight, January 1st, 1970 GMT (the Epoch).

Table 6-1. Notification attributes

Notification attribute	Type	Value range (default value)	Passed by default	Description
Acknowledged	Boolean	TRUE or FALSE	No	TRUE if the notification has been acknowledged, FALSE if not.
Active	Boolean	TRUE or FALSE (Default: TRUE)	No	TRUE if the notification is active, FALSE if not.
AuditTrail	Special		No	Audit trail for the notification. AuditTrail is a table that includes: ■ ActionType ■ SerialNumber ■ Text ■ Timestamp ■ User

Table 6-1. Notification attributes (continued)

Notification attribute	Type	Value range (default value)	Passed by default	Description
Category	String		Yes	Type of notification. Possible values include: ■ Availability ■ Discovery ■ Error ■ Operational ■ Performance ■ PowerSupply ■ Resource ■ Temperature ■ IMPACT These categories correspond to the Exceptions of the same name described in the IP Manager User Guideand the Business Impact Manager User Guide.
Certainty	Float	0 to 100 (Default: 100)	Yes	Confidence level that this notification is the correct diagnosis.
ClassDisplayName	String		Yes	Class name of the managed element (where the event occurred) that is displayed to the user.
ClassName	String		Yes	Class name of the managed element where the event occurred. This attribute along with InstanceName and EventName uniquely identifies this notification. ClassName may differ from the ElementClassName.
ClearOnAcknowledge	Boolean	TRUE or FALSE (Default: FALSE)	Yes	TRUE if the notification should be cleared when it is acknowledged. Use this in cases when notifications never expire or that have sources that will not generate a clear.
ElementClassName	String		Yes	Class Name of the object that the event occurred on.
ElementName	String		Yes	Element Name of the object that the event occurred on.
EventDisplayName	String		Yes	Name of the notification that is displayed to the user.
EventName	String		Yes	Name of the event. This attribute along with ClassName and InstanceName uniquely identifies this notification.

Table 6-1. Notification attributes (continued)

Notification attribute	Type	Value range (default value)	Passed by default	Description
EventState	String		No	Indicates the state of the notification. Possible values include: ■ ACTIVE ■ SUSPENDED ■ WAS_ACTIVE ■ INACTIVE ■ UNINITIALIZED
EventText	String		Yes	A description of the notification.
EventType	Special	MOMENTARY or DURABLE (Default: DURABLE)	Yes	Indicates the nature of the event. A MOMENTARY event has no duration (such as an authentication failure). A DURABLE event has a period during which the event is active and after which the event is no longer active (such as a link failure).
FirstNotifiedAt	Integer	(Default: 0)	No	First notification time. (This value is reset after an event is archived.) Events diagnosed by an underlying domain include timestamps that accurately reflect when the event occurred. The Global Manageruses this value from the event for the FirstNotifiedAt/LastNotifiedAt times when it creates the notification.
Impact	Integer	(Default: 0)	No	A number that quantifies the impact of this notification on the infrastructure or business processes. Larger numeric values indicate a larger impact.
InMaintenance	Boolean	TRUE or FALSE (Default: FALSE)	No	TRUE indicates that the device associated with the notification is in maintenance mode, FALSE if not.
InstanceDisplayName	String		Yes	Name of the instance that is displayed to the user. **Note** If tagging is implemented in the Global Manager, the InstanceDisplayName may include a tag suffix.
InstanceName	String		Yes	Name of the instance where the event occurred. This attribute along with ClassName and EventName uniquely identifies this notification.

Table 6-1. Notification attributes (continued)

Notification attribute	Type	Value range (default value)	Passed by default	Description
IsRoot	Boolean	TRUE or FALSE	No	TRUE if the notification is a root cause, FALSE if not. It is often useful to filter on IsRoot to find those notifications which require immediate attention.
IsProblem	String		Yes	Yes indicates if (and only if) at least one of the source event types is “PROBLEM” and other source event types are either “PROBLEM” or “UNKNOWN”. No indicates that one or more of the source event types is “EVENT” or “AGGREGATE”. Because filtering on IsRoot can result in transient behavior when events are received from multiple underlying sources, it is recommended to filter on IsProblem instead.
LastChangedAt	Integer	(Default: 0)	No	Time since the Epoch when the status of the notification last changed. This is calculated by the Global Manager.
LastClearedAt	Integer	(Default: 0)	No	Time since the Epoch when the notification was last cleared. This is calculated by the Global Manager.
LastNotifiedAt	Integer	(Default: 0)	No	Time since the Epoch when the notification was last notified.
Name	String		No	Internal identifier for the notification.
Occurrence Count	Integer		No	Number of times the notification has occurred.
Owner	String		No	Name of the user responsible for handling this notification.

Table 6-1. Notification attributes (continued)

Notification attribute	Type	Value range (default value)	Passed by default	Description
Severity	Integer	1 to 5 (Default: 5)	Yes	An enumerated value that describes the severity of the notification from the notifier's point of view. Valid values are the integers 1 through 5: 1 Critical indicates action is needed NOW and the scope is broad, for example, an outage to a critical resource. 2 Major indicates action is needed NOW. 3 Minor indicates action is needed, but the situation is not serious at this time. 4 Unknown indicates that the element is unreachable, disconnected, or in an otherwise unknown state. 5 Normal is used when an event is purely informational. **Note** Only the numbers, not the text descriptions, are passed by the Global Manager.
SourceDomainName	String		No	The name of the domain that notified current occurrences of this event. If there is more than one domain, the attribute lists each separated by a comma. An intermediary Managerthat relays an event is not listed as a source. For example, “Trap Processor” (SNMP Trap Adapter) is listed as the source of a notification and not the intermediary SAM Adapter PlatformServer.
SourceEventType	String	UNKNOWN	Yes	Indicates the type of event from an underlying source or sources. Value can be: PROBLEM, EVENT, AGGREGATE, or UNKNOWN. (Not to be confused with Event Type.)
TroubleTicketID	String		No	Trouble-ticket number associated with this notification.
UserDefined1-10	String		No	User defined field 1-10.

VMware Smart Assurance MIB for SNMP Traps

7

The VMware Smart Assurance MIB implemented by VMware Corporation for SNMP Traps is as follows:

```
-- SMARTS-MIB.my: VMware Smart Assurance corporate MIB
--
-- Copyright 1999-2006 by VMware Corporation ("VMware").
-- All rights reserved.
--
-- UNPUBLISHED CONFIDENTIAL AND PROPRIETARY PROPERTY OF VMware.  The
-- copyright notice above does not evidence any actual or intended
-- publication of this software.  Disclosure and dissemination are
-- pursuant to separate agreements.  Unauthorized use, distribution
-- or dissemination are strictly prohibited.
--
-- RCS $Id: SMARTS-MIB.my,v 1.1.38.2 2006/02/06 23:41:15 eo1 Exp $
--
SMARTS-MIB DEFINITIONS ::= BEGIN
IMPORTS
        enterprises, Counter32, OBJECT-TYPE,
        MODULE-IDENTITY, OBJECT-IDENTITY, NOTIFICATION-TYPE
                FROM SNMPv2-SMI;
smartsMIB MODULE-IDENTITY
        LAST-UPDATED "200602070000Z"
        ORGANIZATION "VMware Corporation"
        CONTACT-INFO
                "       Support
                Postal: VMware Corporation
                        Corporate Headquarters
                        176 South Street
                        Hopkinton, MA 01748-9103
                        US
                Phone:
                  United States: (800) 782-4362 (SVC-4VMware)
                  Canada:        (800) 543-4782 (543-4SVC)
                  Worldwide:     (508) 497-7901
 Web: http://powerlink.VMware.com"
                DESCRIPTION
                        "The MIB module for VMware Smart Assurance entities defined by
                        VMware Corporation."
        ::= { enterprises 733 }
-- top level groups in the SMARTS-MIB
-- The smNotificationTrap Group.
```

```
smNotificationTrap OBJECT-IDENTITY
        STATUS  current
        DESCRIPTION
         "This group is acutally the prefix one uses when creating
          enterprise-specific trap OID's for an SNMPv2 trap.  It is
          used in the SMARTS MIBS when defining traps."
        ::= { smartsMIB 0 }
-- The smNotificationData Group.
smNotificationData OBJECT-IDENTITY
        STATUS  current
        DESCRIPTION
         "The members of this group are the OIDs for VarBinds
          containing notification data."
        ::= { smartsMIB 2 }
-- Group for generic notification data.
smGenericNotify OBJECT-IDENTITY
        STATUS  current
        DESCRIPTION
         "The members of this group are the OIDs for VarBinds
          containing generic notification data."
        ::= { smNotificationData 1 }
smNotifTimestamp OBJECT-TYPE
        SYNTAX Counter32
        MAX-ACCESS accessible-for-notify
        STATUS  current
        DESCRIPTION
         "The timestamp of the notification."
        ::= { smGenericNotify 1 }
smNotifServer OBJECT-TYPE
        SYNTAX OCTET STRING
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
         "The name of the server that is sending the notification."
        ::= { smGenericNotify 2 }
smNotifClass OBJECT-TYPE
        SYNTAX OCTET STRING
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
         "The class of the object associated with the notification."
        ::= { smGenericNotify 3 }
smNotifInstance OBJECT-TYPE
        SYNTAX OCTET STRING
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
        "The instance name of the object associated with the notification."
        ::= { smGenericNotify 4 }
smNotifEventName OBJECT-TYPE
        SYNTAX OCTET STRING
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
        "The name of the event causing the notification."
```

```
        ::= { smGenericNotify 5 }
smNotifInstanceID OBJECT-TYPE
        SYNTAX OCTET STRING
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
         "The unique InCharge inventory identification
          for the object associated with the notification."
        ::= { smGenericNotify 6 }
smNotifDescription OBJECT-TYPE
        SYNTAX OCTET STRING
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
         "A complete description of the event."
        ::= { smGenericNotify 7 }
smNotifCertainty OBJECT-TYPE
        SYNTAX OCTET STRING
        MAX-ACCESS accessible-for-notify
        STATUS current
        DESCRIPTION
         "The certainty of the event.  Floating-point number in the
          range 0-100, stored as a string."
        ::= { smGenericNotify 8 }
smNotifSeverity OBJECT-TYPE
        SYNTAX INTEGER {
                notApplicable (1),
                informational (2),
                warning (3),
                minor (4),
                major (5),
                severe (6)
        }
        MAX-ACCESS accessible-for-notify
        STATUS  current
        DESCRIPTION
         "The severity of the event.  Integer number in the range 1-6."
        ::= { smGenericNotify 9 }
-- The SMARTS enterprise Traps
--
-- These are the enterprise-specific trap codes currently in-use in
-- SMARTS software.  The final sub-OID of each object is the code sent
-- in the "specific-trap" field of an SNMPv1 Trap-PDU.
--
-- The definition of these objects mimics the SNMPv2 convention for
-- sending traps:  Take the enterprise OID, append 0, then append the
-- trap code.
-- TRAP NUMBER USE: Trap code 0 is reserved (by SNMP).
--                  Trap codes 1-3 are used by the SMARTS-OV-NV-MIB.
-- TRAPS 4-7: The "smTrap*" series of traps define "base" trap
--            numbers, ones with a generic purpose.  These traps are
--            also used by the generic Trap Adapter.
          smTrapNotification NOTIFICATION-TYPE
            OBJECTS {   smNotifTimestamp,
                        smNotifServer,
```

```
                    smNotifClass,
                    smNotifInstance,
                    smNotifEventName,
                    smNotifInstanceID,
                    smNotifDescription,
                    smNotifCertainty,
                    smNotifSeverity
                }
            STATUS current
            DESCRIPTION
             "A trap describing an InCharge root cause notification.
              The text in smNotifDescription indicates the nature of
              the problem."
          ::= { smNotificationTrap 4 }
          smTrapCertaintyChange NOTIFICATION-TYPE
            OBJECTS {   smNotifTimestamp,
                        smNotifServer,
                        smNotifClass,
                        smNotifInstance,
                        smNotifEventName,
                        smNotifInstanceID,
                        smNotifDescription,
                        smNotifCertainty,
                        smNotifSeverity
                    }
             STATUS current
             DESCRIPTION
              "A trap indicating a certainty change of an InCharge
               notification. The text in smNotifDescription indicates
               the nature of the problem."
           ::= { smNotificationTrap 5 }
           smTrapSeverityChange NOTIFICATION-TYPE
            OBJECTS {   smNotifTimestamp,
                        smNotifServer,
                        smNotifClass,
                        smNotifInstance,
                        smNotifEventName,
                        smNotifInstanceID,
                        smNotifDescription,
                        smNotifCertainty,
                        smNotifSeverity
                    }
             STATUS current
             DESCRIPTION
              "A trap indicating a severity change of an InCharge
               notification. The text in smNotifDescription indicates
               the nature of the notification."
           ::= { smNotificationTrap 6 }
           smTrapNotificationClear NOTIFICATION-TYPE
            OBJECTS {   smNotifTimestamp,
                        smNotifServer,
                        smNotifClass,
                        smNotifInstance,
                        smNotifEventName,
                        smNotifInstanceID,
```

```
                        smNotifDescription,
                        smNotifCertainty,
                        smNotifSeverity
                    }
                STATUS current
                DESCRIPTION
                 "A trap indicating the clear of an InCharge notification."
            ::= { smNotificationTrap 7 }
            smTrapNotificationChange NOTIFICATION-TYPE
             OBJECTS {   smNotifTimestamp,
                        smNotifServer,
                        smNotifClass,
                        smNotifInstance,
                        smNotifEventName,
                        smNotifInstanceID,
                        smNotifDescription,
                        smNotifCertainty,
                        smNotifSeverity
                    }
                STATUS current
                DESCRIPTION
                 "A trap indicating the change of an InCharge notification."
            ::= { smNotificationTrap 98 }
            smTrapNotificationDelete NOTIFICATION-TYPE
             OBJECTS {   smNotifTimestamp,
                        smNotifServer,
                        smNotifClass,
                        smNotifInstance,
                        smNotifEventName,
                        smNotifInstanceID,
                        smNotifDescription,
                        smNotifCertainty,
                        smNotifSeverity
                    }
                STATUS current
                DESCRIPTION
                 "A trap indicating the delete of an InCharge notification."
            ::= { smNotificationTrap 99 }
-- TRAP NUMBER USE: Trap codes 8-99 are reserved for future "base trap number"
--                  use.
--
--                  Trap codes 100-133 are used by SMARTS-PERFORMANCE-MIB.
--                  Trap codes 134-199 are reserved for future use by
--                  SMARTS-PERFORMANCE-MIB.
END
```