CHAPTER 20

Managing Additional Event Types Using the VCB

This chapter contains the following sections:

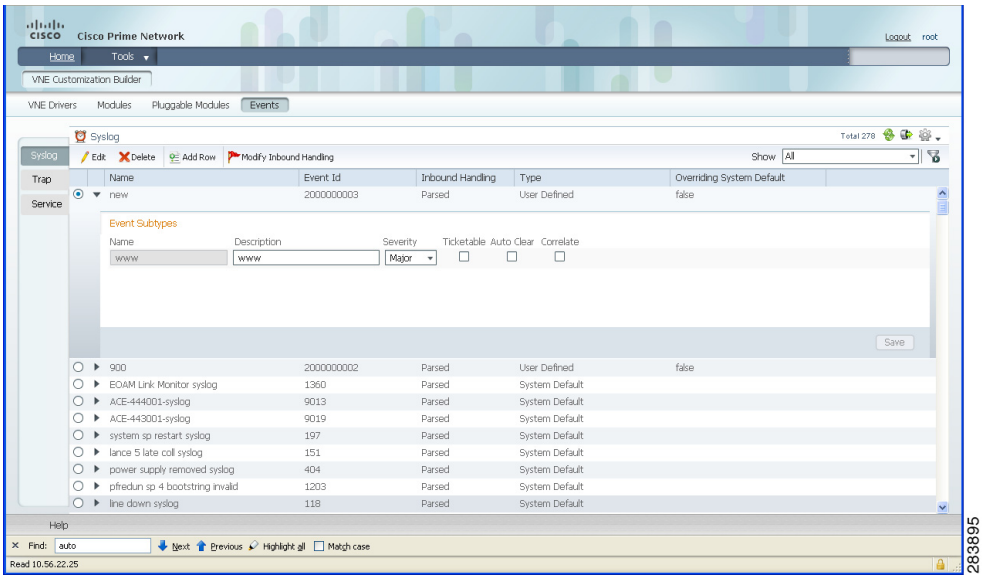
Overview

The VCB can be used to enable Prime Network to recognize additional events and to customize the way events are handled. For example, you might want Prime Network to recognize traps that are specific to a new technology or a custom syslog that you have defined. You might also want to change the settings of a system default event, for example, change the severity from major to minor.

You can customize events using the VCB tool within Prime Network, or using VCB CLI commands. This chapter primarily describes event customization using the Prime Network VCB GUI. For information on using the CLI method, see Using the CLI to Customize Events, page 20-13.

The Events tab in the VCB tool provides all the functionality required to add and customize events.

Figure 20-1 Events Tab



Enabling Support for Additional Events

Using the VCB, you can enable Prime Network to recognize currently unsupported traps or syslogs. The procedures for adding support for events are illustrated through the following example use cases:

Enabling Support for Unsupported Traps, page 20-2

Adding Support for a Custom Syslog, page 20-7

Enabling Support for Unsupported Traps

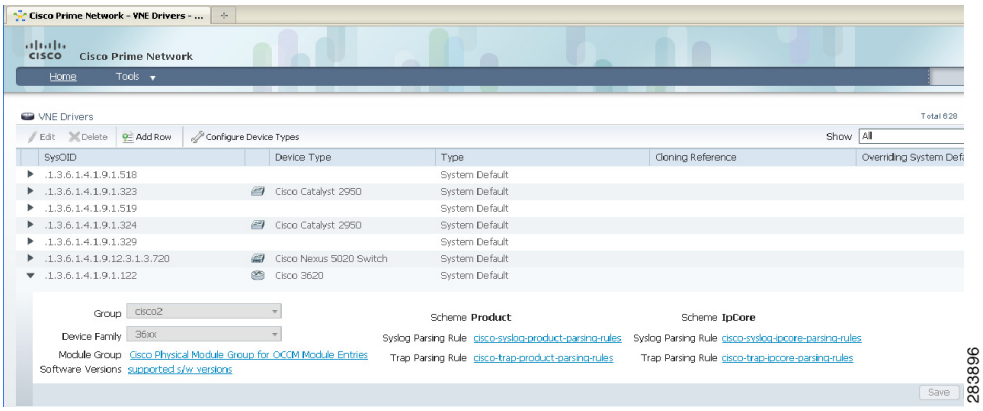
This procedure describes how to add unsupported traps as events in Prime Network based on a particular MIB definition file.

Use the VCB to add support for unsupported traps, as follows:

Step 1 In the VCB tool, go to the VNE Drivers tab.

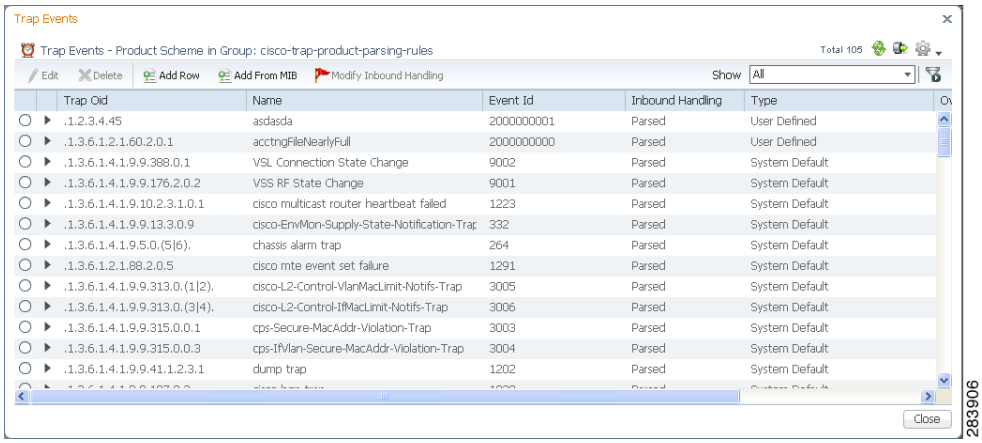
Step 2 Click on the arrow next to the VNE driver on which you want the additional traps to be supported.

Figure 20-2 Expanded VNE Driver Properties Showing Parsing Rules



Step 3 Click the Trap Parsing Rule link to show a list of traps associated with this parsing rule.

Figure 20-3 Traps Associated with cisco-trap-product-parsing-rules

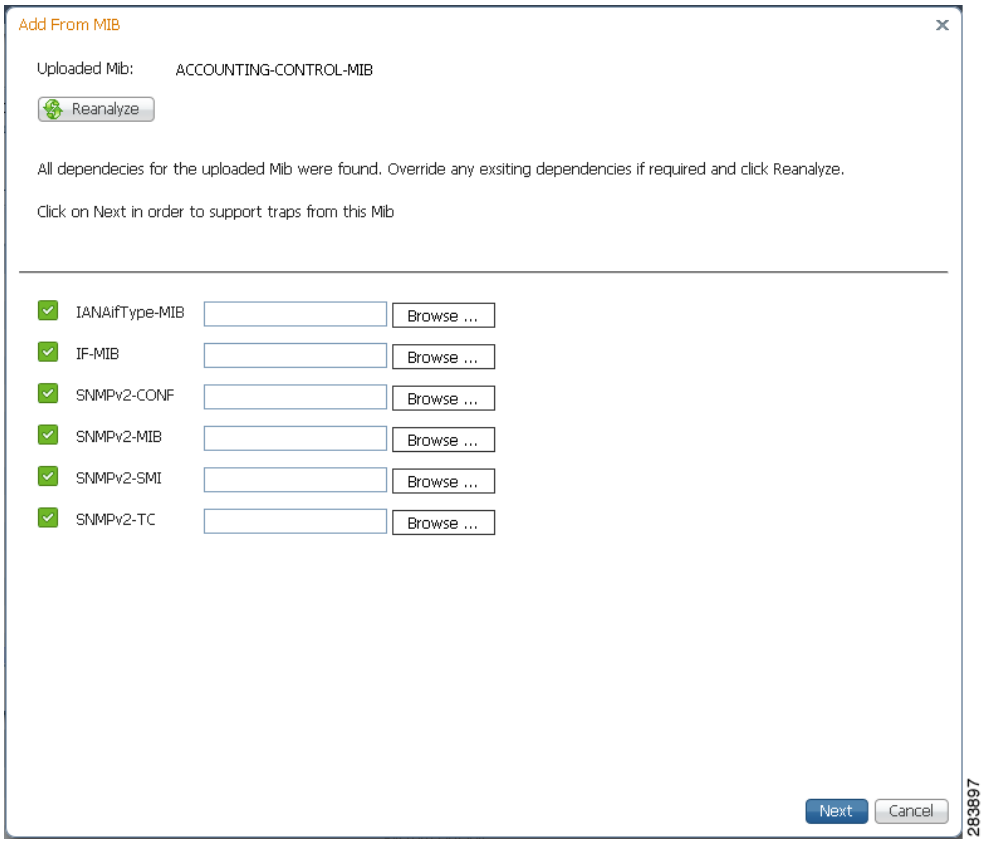


Step 4 Click **Add from MIB**. This launches a wizard which enables you to analyze a specified MIB and select the traps to be supported.

Step 5 Click **Browse** and select the MIB file you want to upload to the Prime Network gateway. The file extension should be .mib or .my or no extension.

A list of MIB dependencies is displayed. A green check mark indicates that the dependency file has been found on the server.

Figure 20-4 MIB Dependencies

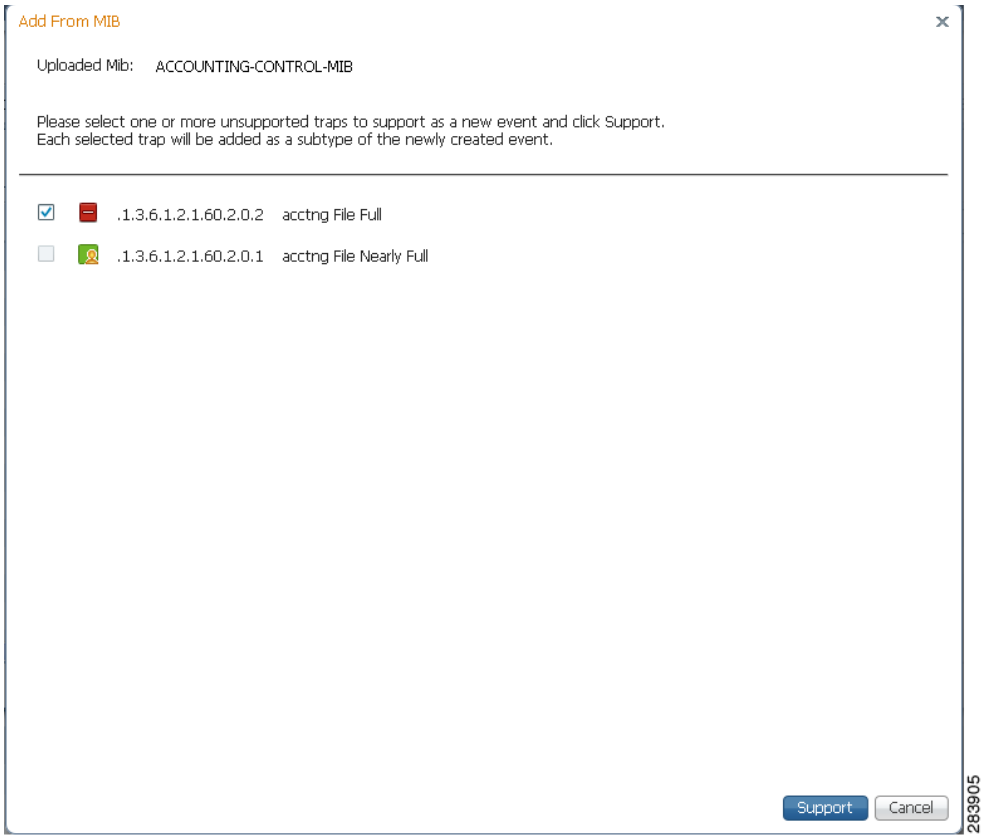


Step 6 If any of the dependencies is not found, click **Browse** and select the dependency file to upload.

Step 7 Click **Next**. A list of traps is displayed. A red icon to the left of the trap indicates that it is not supported. A green icon indicates that the trap is already supported.

Step 8 Check the check box next to the unsupported trap for which you want to add support, and click **Support**.

Figure 20-5 List of Supported and Unsupported Traps



An event is created and the first page of the Event Definition wizard is displayed. The event name and OID are pre-populated but can be changed if necessary. The OID is the common prefix of the OIDs of the selected subtypes.

Step 9 Click **Next** to go to the next step in the wizard which is to define the event subtypes. You will see that a subtype has been created for each unsupported trap you selected. Edit the information for each subtype as required:

- Name—Enter a unique name for the event subtype.
- Description—Enter a string that describes the event subtype.
- Severity—Select the severity to be attributed to the subtype.
- Ticketable—Check the check box if you want Prime Network to create a ticket for this event if there is no root cause event to which it can be correlated.

Note If you make a sub-type ticketable, a ticket will be generated for it. When a non-ticketable sub-type of the same event arrives (for example, a warning or clearing event), the ticket will be updated.

- Auto Clear—Check the check box if you want Prime Network to automatically clear the event. Prime Network clears a ticket if all of its events either are cleared or are configured for automatic clearing.
- Correlate—Check the check box if you want the event to be correlated to a root-cause alarm.

Step 10 Click **Next** to go to the next step in the wizard which is subtype identification. In this step, you define how Prime Network will differentiate between the subtypes, as follows:

- By TrapOID—Select this option if each subtype has a unique OID. In the Replacing Rules section, specify the OID suffix for each subtype. The OID suffix must be an integer.
- By Varbind Value—Select this option if the subtypes have the same trap OID and you want to use one of the varbind values to differentiate between the subtypes. In the Replacing Rules section, select the required varbind from the drop-down menu and then define the values for each subtype.
- By Varbind OID—Select this option if there is a varbind for each subtype. In the Replacing Rules section, specify the common prefix of the varbind OIDs and the suffix for each subtype.

Step 11 Click **Next** to go to the next step in the wizard which is association of the event with the VNE. Specify the following information:

- Source Type—Select the entity to which the event should be associated.
  - ManagedElement Key—Select this option if there is no specific interface or other component of the VNE from which the event is generated.
  - Efp Key From Ifname Serviceid—Associates the event with a specific EFP DC, based on the service instance ID and the interface name.
  - Interface Key From Ifindex—Creates the interface device component key from the ifIndex and associates the event with the appropriate interface layer.
  - Interface Key From Ifname—Associates the event with a specific interface that you specify in the Interface Identifier field.
  - Logical Container Key—Associates the event with a designated logical container that you select in the Logical Container field.
  - ModuleDC Key Given EntPhysicalIndex—Associates the event with a designated module DC.
  - ModuleDC With SlotSubslot Value Key—Associates the event with the corresponding module, based on the slot number.
  - Pw Interface Key From Tunnelindex—Associates trap events with the designated pseudowire tunnel interface.
- Logical Container—Applicable only when the source type is Logical Container Key. This field lists the various logical containers for which the VCB supports event association. For example, BGP traps/syslogs can be associated with the MP-BGP type container, ISIS events with the ISIS System container, and so on.
- Instance Identifier Location—Specify whether the identifier of the event is based on a value or a varbind OID.
- Instance Identifier Varbind OID—Select the varbind that contains the instance information. Prime Network uses varbind OID to locate the varbind in the trap PDU and locates the instance information from either the OID or the value depending what was selected as the instance identifier location (OID or value).
- Source Location—Specify whether the event source can be found in a value or a varbind OID.
- Source Varbind ID—Select the varbind that contains the source information. Prime Network uses varbind OID to locate the varbind in the trap PDU and locates the source information from either the OID or the value depending what was selected as the source location (OID or value).

Step 12 Click **Next** to go to the next step in the wizard which determines which VNE drivers will be extended to support this event. This is determined by selecting the parsing rule groups per scheme that will be extended. The event will be supported on all VNE drivers that use the specified parsing rule groups.

Step 13 Click **Add** to select additional parsing rule groups, as required. You can select additional groups for the Product scheme or you can select the IpCore scheme and a parsing rule group.

Note Certain parsing rules groups inherit from other groups. If you select multiple groups, make sure that your selection does not include a base (parent) group as well as the group that inherits from the base group. See Parsing Rule Group Inheritance Structure, page 20-10 for the relationship between parsing rule groups.

Note For changes to take effect, you must restart the VNEs that are affected.

Adding Support for a Custom Syslog

In this example, a custom syslog is generated by a router, using Embedded Event Manager (EEM), when the Windows XP server being monitored is not reachable. The custom syslog is %HA_EM-6-LOG: IPSLA-XP: Windows-XP unreachable. This event is sent to Prime Network but is not recognized or parsed.

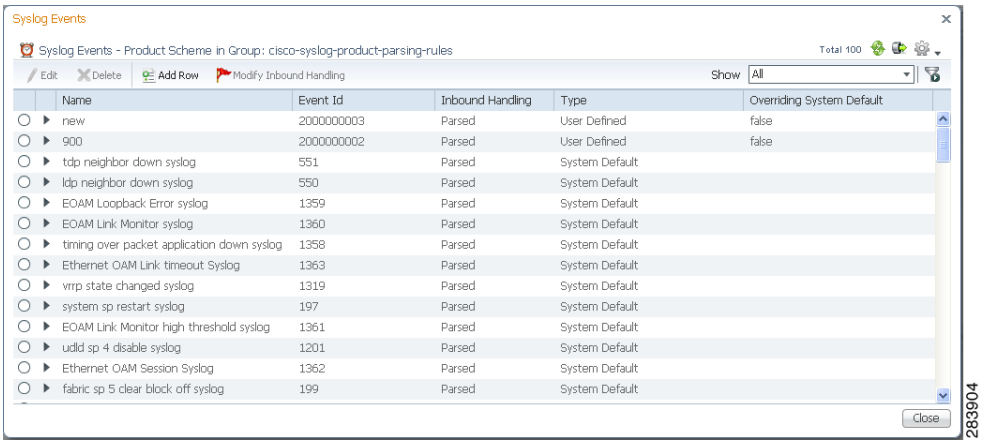
Use the VCB to add support for this custom syslog, as follows:

Step 1 In the VCB tool, go to the VNE Drivers tab.

Step 2 Click on the arrow next to the VNE driver that represents the router that generates the custom syslog to expand its display. The Syslog Parsing Rule field shows the parsing rule used to parse events for this VNE driver, for both Product and IpCore schemes.

Step 3 Click the Syslog Parsing Rule link to show a list of syslog events associated with this parsing rule.

Figure 20-6 Syslogs Associated with syslog-product-parsing-rules



Step 4 Click the Add Row button to start defining the new syslog.

Step 5 Enter a unique name for the syslog in the Event Name field. For example, Monitoring XP Server.

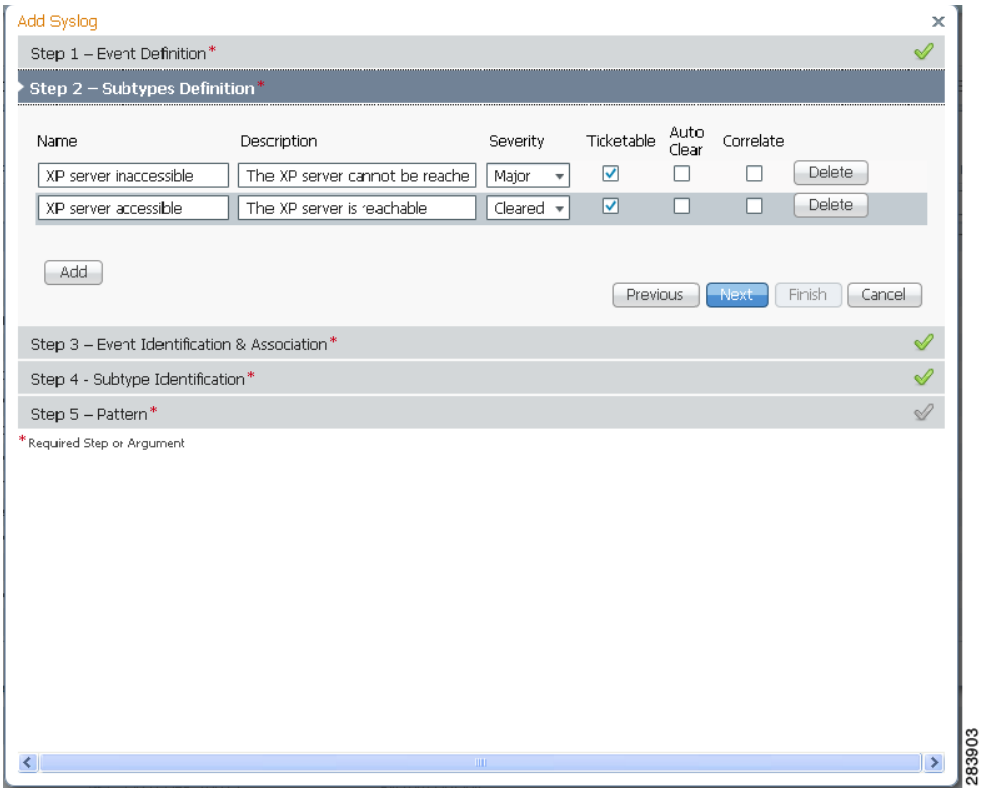
Step 6 Click **Next** to go to the next step in the wizard which is to define the event subtypes.

Step 7 Enter the following information to define the first event subtype:

- Name—enter a unique name for the event subtype, for example, XP server inaccessible.
- Description—enter a string that describes the event, for example, "The XP server cannot be reached."
- Severity—Select the severity to be attributed to the event.
- Ticketable—Check the check box if you want Prime Network to create a ticket for this event if there is no root cause event to which it can be correlated.
- Auto Clear—Check the check box if you want Prime Network to automatically clear the event, without waiting for a clear event or for manual clearing of the event. If the auto clear check box is checked, the event will be cleared automatically 4 minutes after the last modification.
- Correlate—Check the check box if you want the event to be correlated to a root-cause alarm.

Step 8 Click **Add** to define a second subtype, for example, XP server accessible, with severity "Cleared".

Figure 20-7 Subtype Definition



Step 9 Click **Next** to go to the next step in the wizard which is identification and association of the event. In this step, you will provide an example of the raw event and you will define parameters by which the event will be identified.

Step 10 Enter the following information to define event identification and association:

- Raw Event—Provide the raw event syntax as an example, so that the system can parse it.
- Subtype Key—Keyword that identifies the subtype. The keyword should be taken from the raw event. In this example, the key would be "unreachable".

- Source Type—Select the source component of the device from which the event is generated. For this example, select ManagedElement Key because there is no specific interface or other component from which the event is generated. In other cases, you might choose from the following options:
  - Efp Key From Ifname Serviceid—Associates the event with a specific EFP DC, based on the service instance ID and the interface name.
  - Interface Key From Ifname—Associates the event with a specific interface that you specify in the Interface Identifier field.
  - Logical Container Key—Associates the event with a designated logical container that you select in the Logical Container field.
  - ModuleDC With SlotSubslot Value Key—Associates the event with the corresponding module, based on the slot number.
- Logical Container—Applicable only when the source type is Logical Container Key. This field lists the various logical containers for which the VCB supports event association. For example, BGP traps/syslogs can be associated with the MP-BGP type container, ISIS events with the ISIS System container, and so on.
- Interface Identifier—Specify a value by which the interface will be identified (ifIndex is used to identify the interface).
- Interface Name—Specify the name of the interface to which the event is associated ((ifName is used to identify the interface).
- Instance Identifier—Specify the identifier of the instance. For this example, the instance identifier could be Windows XP.

Step 11 Click **Next** to go to the next step in the wizard which is identification of the event subtypes. In this step, you will define values for each of the subtypes. In this example, the values could be “unreachable” for XP server inaccessible and “reachable” for XP server accessible.

Step 12 Click **Next** to go to the next step in the wizard which determines which VNE drivers will be extended to support this event. This is determined by selecting the parsing rule groups per scheme that will be extended. The event will be supported on all VNE drivers that use the specified parsing rule groups.

Step 13 Click **Add** to select additional parsing rule groups, as required. In this example, all VNE drivers associated with group cisco-syslog-product-parsing-rules will be extended to support the new syslog. You can select additional groups for the Product scheme or you can select the IpCore scheme and a parsing rule group.

Note Certain parsing rules groups inherit from other groups. If you select multiple groups, make sure that your selection does not include a base (parent) group as well as the group that inherits from the base group. See Parsing Rule Group Inheritance Structure, page 20-10 for the relationship between parsing rule groups.

Step 14 Click **Finish**. The event now appears in the list of syslogs for the cisco-syslog-product-parsing-rules group.

Parsing Rule Group Inheritance Structure

Table 20-1 Parsing Rule Groups Inheritance

Parsing Rule Group	Inherits From.....
cisco-asr90xx-syslog-ipcore-parsing-rules	cisco-iox-syslog-ipcore-parsing-rules[1]
cisco-asr90xx-trap-ipcore-parsing-rules	cisco-iox-trap-ipcore-parsing-rules
cisco-ciscocpt-trap-ipcore-parsing-rules	cisco-trap-ipcore-parsing-rules
cisco-ciscocpt-trap-product-parsing-rules	cisco-trap-product-parsing-rules
cisco-iox-syslog-ipcore-parsing-rules	cisco-iox-syslog-product-parsing-rules
cisco-syslog-ipcore-parsing-rules	cisco-syslog-product-parsing-rules
cisco-trap-ipcore-parsing-rules	cisco-trap-product-parsing-rules
nexus-trap-product-parsing-rules	nexus-422v-trap-product-parsing-rules

1. Multiple inheritance levels: cisco-iox-syslog-ipcore-parsing-rules inherits from cisco-iox-syslog-product-parsing-rules. Therefore, if cisco-iox-syslog-product-parsing-rules is selected, do not select cisco-asr90xx-syslog-ipcore-parsing-rules and cisco-iox-syslog-ipcore-parsing-rules.

Customizing Events

Using the VCB, you can change the way Prime Network deals with events. For example, you can change the severity of an event, or you can instruct the system to drop the event. Event customization is described in the following sections:

- Changing the Severity of an Event Subtype, page 20-10
- Dropping an Event, page 20-11
- Restoring a System Default Event, page 20-11
- Deleting Events, page 20-12

Changing the Severity of an Event Subtype

The events that are supported by default in Prime Network are attributed with a specific severity. You can customize the event and change the severity if it is not appropriate for your organization. For example, the Prime Network system considers the event, “ASR5 port down” to have a severity of “Warning”. However, in your organization, this event might be considered to be “Major” and you want the event to be marked as such.

To change the severity of an event subtype:

Step 1 In the VCB tool, select **Tools > Events** or click on the Events tab.

Step 2 Click the arrow next to the event you want to customize to expand its display.

Step 3 Select the required severity from the Severity drop-down menu and click **Save**.

Dropping an Event

By default, when an event is received by Prime Network, it is archived and parsed. Only events that have been parsed will appear in Prime Network Events tables. You can choose to drop an event so that it no longer appears in the tables. The event will no longer be actionable, meaning that it will not be processed and parsed, but it will be archived. In the case of service events, the event will no longer be generated by the system so there will be no archiving.

To drop an event:

Step 1 In the VCB tool, select **Tools > Events** or click on the Events tab.

Step 2 Select the event you want to drop and click **Modify Inbound Handling**.

Step 3 Click **OK** in the confirmation message.

The Inbound Handling column for the event will change to Archived Only for syslogs and traps or to Disabled for system events.

Restoring a Dropped Event

To restore a dropped event:

Step 1 In the VCB tool, select **Tools > Events** or click on the Events tab.

Step 2 Select the event you want to drop and click **Modify Inbound Handling**.

Step 3 Click **OK** in the confirmation message.

The Inbound Handling column for the event will change to Archived Only for syslogs and traps or to Disabled for system events.

Restoring a System Default Event

If you have edited a system default event and you want to go back to the original event, you can restore the system default event.

The Overriding System Default column indicates whether or not a system default event has been edited. The values for this column are true or false.

Note A VNE upgrade package might provide support for events that you previously added using the VCB. After you have upgraded the VNE driver, such events are marked as overriding the system default. Use this procedure to restore the system default event that is provided with the upgrade.

To restore a system default event:

Step 1 In the VCB tool, select **Tools > Events** or click on the Events tab.

Step 2 Select the event you want to restore and click **Restore**.

Step 3 Click **OK** in the confirmation message.

The Overriding System Default column for the event will change to False.

Deleting Events

User-defined events can be deleted as long as they are not overrides of system default events. System default events cannot be deleted.

Note For system default overrides, a Restore button is provided instead of the Delete button.

To delete an event:

Step 1 In the VCB tool, select **Tools > Events** or click on the Events tab.

Step 2 Select the event you want to delete and click **Delete**.

Step 3 Click **OK** in the confirmation message.

The event is removed from the table.

Testing and Certifying Event Customizations

It is recommended that you test and certify event customizations in your lab before moving them production:

To prepare your test environment:

Step 1 Restart Prime Network or restart at least all VNEs/AVMs that need to support the new event. A restart is required because event patterns are loaded upon VNE initialization only.

Step 2 If using a simulator, enable Prime Network to process traps sent from a simulator by running this command and restarting avm100:

```
runRegTool.sh -gs localhost set 0.0.0.0
avm100/agents/trap/processors/snmp-processor/class
com.sheer.metrocentral.framework.instrumentation.trap.processor.RawAgentIpSnmpEvent
Processor
```

Step 3 To receive events from a device, you must configure the device with the details of the Prime Network server. For example, use the following commands for devices running Cisco IOS or Catalyst OS software:

```
snmp-server host 172.20.2.160
logging trap informational
logging source-interface Loopback0
logging on
logging 172.20.2.160
```

A similar set of commands should be used for devices belonging to other manufacturers.

These commands enable the device to send traps to the specified gateway IP over port 162. Therefore, port 162 must be enabled to receive traps from the device. In addition, you must reserve port 1162 for the general trap processing by the Prime Network server. This task is handled by the AVM 100 process.

To test your event:

Step 1 Send an event from an NE or from a simulator. Open Prime Network Events and verify that:

- Prime Network recognizes and processes the event
- Event parameters—type, subtype, severity, and so on—are as expected
- Unique ID is appended to source ID (ManagedElement)

After you complete the tests, move the customization to production and test there as well; see Moving a Tested and Certified Event Customization to Production, page 20-13.

Moving a Tested and Certified Event Customization to Production

Tip Always perform customization during a scheduled maintenance window.

Export the configuration from your lab setup and import it into the production setup using the following procedures:

- Exporting VCB Registry Customizations, page 21-3
- Importing VCB Registry Customizations, page 21-3

After you import the changes to the production server:

1. Restart Prime Network or at a minimum, restart all VNEs and AVMs that need to support the new events.
2. Repeat testing and certification to ensure that the customizations are functioning as expected in your production environment. See Testing and Certifying Event Customizations, page 20-12

Using the CLI to Customize Events

You have the option to use the CLI to perform VCB customizations. Use the CLI if you are an advanced VCB user or if the current VCB GUI does not support the customization you need to perform.

For information about general VCB commands, see Introductory Command Reference and Global Options, page 18-20.

This section describes the CLI commands that can be used to customize events, as follows:

- vcb event Commands, page 20-14
- vcb eventparsingrules Commands, page 20-21
- vcb eventpattern Commands, page 20-28
- vcb eventarg Commands, page 20-33

vcb event Commands

Use the following **vcb event** commands to create, view, modify, and delete events:

vcb event add

Use the **vcb event add** command to create an event definition for a syslog or a trap in Prime Network based on user input.

Note To create a script to add unsupported traps from a MIB, use the e **vcb event view** command with the **-generatecli** option.To list the traps in a MIB that are supported and those that are not, use the **vcb event view** command with the **-mibfile** option. For more information, see vcb event view, page 20-16.

Synopsis

```
vcb event add -eventtype {syslog|trap} -eventname eventName [-alarmid alarmId]
{-subtype1 subtype1Name [-ticketable1]
[-severity1 critical|major|minor|warning|info|cleared>]
[-shortdesc1 short description string] [-autoclear1 false|true]
. . .
[-subtypen subtypenName] [-ticketablen][-severityn
critical|major|minor|warning|info|cleared ]
[ -shortdescn short description string] [-autoclearn false|true]

-user username -password password
```

Description

The **vcb event add** command creates an event definition in Prime Network based on the user input. Afterwards, the VNE-driver can create specific instances of this event for incoming traps or syslogs, persist them in the event database, and forward them to interested clients.

Usage Example

```
vcb event add -eventtype syslog
-eventname "stack switch status syslog"
-subtype1 "stack switch removed syslog"
-severity1 minor
-subtype2 "stack switch added syslog"
-severity2 cleared
-user root -password admin
```

Adds a syslog event definition with the name “stack switch status syslog”. Two subtypes are added: “stack switch removed syslog” with severity minor and “stack switch added syslog” with severity cleared. By default, the subevents are not ticketable.

The syslog for which the event definition was added is:

```
STACKMGR-4-SWITCH_[ADDED|REMOVED]: Switch [dec] has been [ADDED to|REMOVED from] the
stack.
```

A device sends one syslog when a switch is added to a stacked device (clear alarm) and another syslog when a switch is removed from a stacked device (asserted minor alarm).

Options

Table 20-2 Options and Arguments—vcb event add

Option *Argument*	Description
eventtype *event type*	Type of event. Valid values are: • syslog • trap
eventname *event name*	Unique string that identifies the event within Prime Network.
alarmid *alarm ID*	(Optional) Unique integer identifier for the event. **Note** Recommendation—Do not provide this argument; the VCB automatically generates a unique number.
subtype$_n$ *subtypen name*	Unique string that identifies the subevent with the Prime Network event.
ticketable$_n$	(Optional) Optional parameter for subtype. If specified, indicates that a ticket should be generated for this subevent. By default, no ticket is generated for the subtype.
-autoclear$_n$ `false|true`	(Optional) Optional parameter for subtype. If the event is ticketable, setting autoclear to false causes the subevent to remain asserted until the clear alarm arrives or the user manually acknowledges or clears the subevent. **Note** Root cause events are not autocleared even when autoclear is set to false. For information about root cause events, see "Fault Management" and "Causality Correlation" in *Cisco Active Network Abstraction 3.7.2 Theory of Operations*. By default, autoclear is true for user-defined event definitions.
severity$_n$ *severity level*	Mandatory parameter for subtype. The severity of the subevent. Possible values are critical, major, minor, warning, info, and cleared.
shortdesc$_n$ *short description*	Optional parameter for subtype. A short description of the subevent. This string is stored in the event database. By default, the subtype value is used as the default shortdesc value.

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-3 Error Codes—vcb event add

Code	Description
201	Event name already exists in Prime Network.
202	Alarm ID already exists in Prime Network.

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb event view

Use the **vcb event view** command to list event definition registrations.

Synopsis

vcb event view -eventname {*eventName* | *all*} [**-substringmatch**]} **-user** *username* **-password** *password* **-*eventtype*** {*trap* | *syslog* | *service*}

vcb event view -genericevents *all* | *trap* | *syslog* [**-ipaddress** *vneip*] [**-date** *yyyy-mm-dd*] [**-time** *hh:mm:ss*] [**-maxrecords** *num*]**-user** *username* **-password** *password*

vcb event view -mibfile *complete-path-mibFilename* [**-generatecli -repository** *ParsingrulesHive* **- group** *PatternsHive*] **-user** *username* **-password** *password*

Description

The **vcb event view** command enables you to view event definitions including event properties such as alarm ID, event subtypes, severity, and ticketability.

Usage Examples

vcb event view -userdefined -eventname *all* **-user** *root* **-password** *admin*

Returns all the event definitions that were added to Prime Network using the VCB.

vcb event view -eventname *bgp* **-substringmatch -user** *root* **-password** *admin*

Returns all BGP event definitions in Prime Network, including those that were added using the VCB.

vcb event view -user *root* **-password** *admin* **-eventname** *bgp* **-substringmatch -eventtype** *service*

Returns all BGP service events.

vcb event view -mibfile */mibs/IF-MIB* **-user** *root* **-password** *admin*

Returns lists of supported events and unsupported events based on the traps in the IF-MIB file.

vcb event view -mibfile */mibs/IF-MIB* **-generateeventcli -group** *cisco-trap-product-parsing-rules* **-repository** *cisco-trap-repository* **-user** *root* **-password** *admin*

Creates, but does not run, a script /Main/VcbEventCommand.sh. The script contains three **vcb** commands for each unsupported trap; the commands add an event (and provide an event ID), event parsing rules, and an event pattern. Optionally, edit the script. To run the script, change permissions on the file to ensure that it is executable and supply a username and password as input; see this example:

```
chmod 755 VcbEventCommand.sh
```

```
./VcbEventCommand.sh -user root -password admin
```

Note The Prime Network gateway maintains a known list of MIBs that are used to provide translation for trap varbinds when displayed in the UI. When an event is added from a MIB that is unknown to the gateway, the VCB does not add the MIB to the known MIB list. As a result, the varbinds for this trap might not be translated to user-friendly names.

Options

Table 20-4 Options and Arguments—vcb event view

Option *Argument*	Description
eventname *eventName*	Unique string that represents the event. **Tip** Enter **all** as the *eventName* to display information on all the event definitions in Prime Network. Use this argument with caution because the number of events can potentially be very large.
substringmatch	(Optional) Indicates that the event name argument is not an exact match.
eventtype *trap* | *syslog* | *service*	Displays events of a specific type - traps, syslogs or service events.
genericevents *generic event type*	Displays all events from the Prime Network database (in raw format). **Tip** Enter **all** as the *generic event type* to display information on all the events in Prime Network.
ipaddress *neip*	(Optional) Generic events filter. IP address for the NE for which you want to see generic events.
date *yyyy-mm-dd*	(Optional) Generic events filter. The date after which the events arrived.(Returns events that arrived after the given day.)
time *hh:mm:ss*	(Optional) Generic events filter. The time after which the events arrived. (Returns events that arrived after the given time.)
maxrecords *num*	(Optional) Generic events filter. The maximum number of events that you want to display. The default value is 100.
mibfile *complete-path-mibFilename*	Loads MIB modules and compares the traps defined in the MIB against the events that are supported in Prime Network. Displays lists of supported traps and unsupported traps. **Note** Before using this command option, copy the MIB and dependent MIB files to a local folder. Rename each MIB file, removing the .my file extension from it.
generatecli	(Optional) When provided, produces a script, *NETWORKHOME*/Main/VcbEventCommand.sh. The script contains commands to add basic event support for each unsupported trap that was identified through the **-mibfile** option.

Table 20-4 Options and Arguments—vcb event view (continued)

Option *Argument*	Description
repository *ParsingrulesHive*	Mandatory **-generatecli** option. Hive that includes event parsing rules for traps.
group *PatternsHive*	Mandatory **-generatecli** option. Hive that includes event patterns for traps.

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-5 Error Codes—vcb event view

Code	Description
231	No such event exists in the events file

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb event modify

Use the **vcb event modify** command to modify events that were previously defined using the VCB or to modify event attributes for factory-defined events (by using the **-override** option). This command can also be used to drop events.

Synopsis

```
vcb event modify -eventname eventName [-alarmid alarmId] [-override]
{-subtype1 subtype1Name {[-ticketable1] [-autoclear1 false|true]-severity1
critical|major|minor|warning|info|cleared
-shortdesc1 short description string…
{-subtypen subtypenName [-ticketablen] [-autoclear1 false|true]
-severityn critical|major|minor|warning|info|cleared
[-shortdescn] short description string -user username -password password
[-eventtype {trap | syslog | service | drop}
```

Description

The **vcb event modify** command modifies an event definition in Prime Network. It can also be used to instruct the system to drop a specific event.

Note Support for modifying an event is limited due to the complexity involved. When additional changes are required—such as changing the name of an event or a subtype—the supported procedure is to delete the entire event definition and add it afresh:

- Delete the event, event pattern, and associated event parsing rules.

- Add the event, event pattern, and event parsing rules.

Usage Examples

```
vcb event modify -eventname "stack switch status syslog"
-subtype1 "stack switch removed syslog" -severity1 major -ticketable1 -user root
-password admin
```

Updates the event definition for the stack switch status syslog, changing the severity of the specified subtype to major and making the subtype ticketable. For a corresponding example of how this event was added, see Usage Example, page 20-14 for the **vcb event add**.

```
vcb event modify -eventname "bgp trap" -override -eventtype drop -user root -password admin
```

Drops the “bgp trap” event. This overrides the system-defined event.

Options

Table 20-6 Options and Arguments—vcb event modify

Option *Argument*	Description
eventname *eventName*	Unique string identifies the event within Prime Network.
eventtype *trap* | *syslog* | *service* | *drop*	Modifies an event of a specific type or instructs the system to drop an event.
alarmid *alarmId*	(Optional) Unique integer identifier for the event. If not provided, the VCB automatically generates a unique number. **Note** We recommend that you do not provide an input alarm ID.
subtype*n* *subtypenName*	Unique string that identifies the subevent with the Prime Network event. To retain ticketability for any ticketable subtype—whether you want to modify the subtype or not —you must enter the subtype option and argument along with the ticketable option (below).
ticketable*n*	(Optional) Parameter for subtype. Indicates whether a ticket should be generated for this subtype. If not specified, no ticket is generated. **Note** To retain ticketability, supply the ticketable option for all subtypes that are currently defined as ticketable events (even for subtypes that you do not intend to modify). Otherwise, the subtypes are modified to be non-ticketable events.

Table 20-6 Options and Arguments—vcb event modify (continued)

Option *Argument*	Description
-autoclear$_n$ `false|true`	(Optional) Parameter for subtype. If the event is ticketable, setting autoclear to false causes the subevent to remain asserted until the clear alarm arrives or the user manually acknowledges or clears the subevent. **Note** Root cause events are not autocleared even when autoclear is set to false. For information about root cause events, see "Fault Management" and "Causality Correlation" in *Cisco Active Network Abstraction 3.7.2 Theory of Operations*. By default, autoclear is true for user-defined event definitions.
severity*n value*	(Optional) Parameter for subtype. Specifies the severity of the subevent. Possible values are critical, major, minor, warning, info, and cleared.
shortdesc*n* `short description string`	(Optional) Parameter for subtype. A short description.
override	(Optional) Indicates that you expect to override attributes for a factory-defined event.

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-7 Error Codes—vcb event modify

Code	Description
202	Alarm ID already exists in Prime Network.
251	Event name does not exist in Prime Network.
252	Event subtype name does not exist for the event.

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb event delete

Use the **vcb event delete** command to delete an event definition. The **vcb event delete** does not delete or change the event template from which the event definition was cloned.

Synopsis

```
vcb event delete -eventname eventName -user username -password password
```

Description

The **vcb event delete** command removes event definitions created using the VCB and removes event attribute overrides from factory-defined events. Deleting a factory-defined event removes event attribute overrides only and not the event itself; the original event attributes are then applied to future events.

Usage Example

```
vcb event delete -eventname "stack switch status syslog" -user root -password admin
```

Deletes the event definition. All registry entries added as a part of the event add command are removed from the site.xml file.

Options

Table 20-8 Options and Arguments—vcb event delete

Argument	Description
eventname *eventName*	Name of the event to be deleted

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-9 Error Codes—vcb event delete

Code	Description
231	No such event exists in the events file

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb eventparsingrules Commands

Use the following **vcb eventparsingrules** commands to create, view, modify, and delete event parsing rules:

vcb eventparsingrules add

Use the **vcb eventparsingrules add** command to create a VNE-driver registration for adding parsing rules to support a new trap or syslog by customizing a specified set of event templates.

Synopsis

```
vcb eventparsingrules add -templates templateName1, templateName2, …, templateNamen
-group repository -rulename rulename [-enable]
{ [-arg1 -arg1Value…-argN argNValue] } -user username -password password
```

Description

The **vcb eventparsingrules add** command creates a VNE-driver event registration based on the templates chosen by the user. This enables Prime Network to identify and associate the event to a particular device component instead of classifying the event as a generic event.

The command does the following:

- Creates a separate registry configuration (a copy) for customizing the event. Parameters that you input using the command affect the copy.
- Creates rules for handling the event based on the event template and user input.
- Updates the site.xml file, so that Prime Network can differentiate customizations created using the VCB from changes supplied in VNE-driver registration files.

Usage Example

```
vcb eventparsingrules add
   -templates
syslog-identification,syslog-subtype-from-expression,create-managedelement-key,create-a
na-syslog-event
   -group cisco-syslog-repository
   -rulename stack-switch-status-syslog
   -syslog_identification_testmessage "STACKMGR-4-SWITCH_ADDED: Switch 2 has been added
to the stack"
   -syslog_identification_expression "STACKMGR-4-SWITCH_$$subtypekey$$: Switch
$$uniqueid$$ has been .*"
   -syslog_subtype_from_expression_replacing_rules "ADDED-stack switch added
syslog,REMOVED-stack switch removed syslog"
-create_ana_syslog_event_type "stack switch status syslog"
-user root -password admin
```

Adds parsing rules to identify the syslog correctly, associates it with the correct device component, and creates the corresponding Prime Network event and subevent.

Four event templates are entered:

- syslog-identification—Rules that pertain to syslog identification.
- syslog-subtype-from-expression—Rules that map the syslog values (in this case, ADDED and REMOVED) to the event subtype names: stack switch added syslog, stack switch removed syslog. The example includes two rules, separated by commas.

Note Rules must be comma-separated. Each rule must include a value and event subtype, separated by a hyphen: value-event subtype.

- create-managedelement-key—Indicates that the syslog should be associated with the managed element. (There are no input parameters for this template.)
- create-ana-syslog-event—Provides rules for creating instances of the corresponding Prime Network event (defined with the **vcb event add** command).

Input parameters for the event templates are variable arguments that depend on the templates selected.

- syslog_identification_expression—The actual syslog message with input that is of interest to the user and is masked with special keys, such as %%subtypekey%%, %%uniqueid%%, and %%entityid%%, depending on which is applicable. In the previous example, only subtypekey and uniqueid parameters are relevant.

Note Only a substring of the message is used in the example, because whatever comes afterward is of no interest to the user.

- syslog_subtype_from_expression_replacing_rules—Specifies the mapping from the subtypekey to the subevent name. The subevent string should exactly match one of the subevent names that was defined using the **vcb event add** command.
- create_ana_syslog_event_type—Specifies the event name.

Note This parameter should exactly match the event name defined using the **vcb event add** command.

Options

Table 20-10 Options and Arguments—vcb eventparsingrules add

Option *Argument*	Description
templates *template name1, ... template namen*	Comma-separated list of event template names. Event templates are divided into categories that correspond to the function they fulfill (identification, association, and so on.) Depending upon the trap or syslog that you are adding, select no more than one template from each template category.
group *repository*	Specifies the vendor-specific trap or syslog repository file under which the customizations should be made.
rule *rulename*	String that is used as a key name for event rule definition.
enable	(Optional) Indicates whether the rule is enabled or disabled. Only enabled rules are used to parse incoming traps and syslogs.
variable arguments	Arguments vary from one event template to another event template.
syslog_identification_ testmessage	(Optional) Parameter valid for syslogs only. An example syslog message used to check the correctness of the regular expression that the VCB creates automatically based on user input.

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-11 Error Codes—vcb eventparsingrules add

Code	Description
211	Event template file not found.
103	No such template name in template file.
212	Only one template can be selected from each template category.
213	Invalid expression for syslog.

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb eventparsingrules view

The **vcb eventparsingrules view** command displays event registrations. Use it to verify that you successfully added an event parsing rule or to view parameters (to fill them in based on the example).

Synopsis

```
vcb eventparsingrules view -group repository name -rulename { rulename | all } [-detail]
vcb eventparsingrules view -template templateName | all -inputparam -user username
-password password
```

Description

The **vcb eventparsingrules view** command shows configuration settings. Use it to list:

- Details of the events repository for a particular rulename or for all the events in the file. The information displayed includes the parsing rules and important parameters in each rule.
- Input parameters in the event template specified by -template option. If all is specified, the user input parameters of all the templates are displayed.

Usage Examples

Example 1

```
vcb eventparsingrules view -template syslog-identification -inputparam -user root
-password admin
```

Displays the syslog-identification event template definition, including a detailed description of the input parameters required when using the template to add event parsing rules.

Example 2

```
vcb eventparsingrules view -group cisco-syslog-repository -userdefined -rulename all
-user root -password admin
```

Displays all event parsing rules that were defined using the VCB under the cisco-syslog-repository hive.

Example 3

```
vcb eventparsingrules view -group cisco-syslog-repository
-rulename stack-switch-status-syslog -user root -password admin
```

Displays the event parsing rules for the event "stack-switch-status-syslog" which was created using the VCB.

Example 4

```
vcb eventparsingrules view -group cisco-syslog-repository -rulename all -user root
-password admin
```

Displays all the event parsing rules present in the hive cisco-syslog-repository, including those added using the VCB.

Options

Table 20-12 Options and Arguments—vcb eventparsingrules view

Option *Argument*	Description
group *repository name*	The trap or syslog repository filename.
rulename *ruleName*	The unique string that is used to represent the event parsing rules. **Tip** Enter **all** as the *ruleName* to display information on all the rules in Prime Network.
detail	(Optional) Lists the entire rule contents including the parsing rule entry details.
template *templateName*	The event template name. **Tip** Enter **all** as the *templateName* to display information on all event templates in Prime Network.
inputparam	(Optional) Lists template definition entries that require user input when creating event parsing rules.

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-13 Error Codes—vcb eventparsingrules view

Code	Description
103	No such template name in the templates file.
222	Parsing rules repository not found.
231	No such rule name in the site.xml.

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb eventparsingrules modify

Use the **vcb eventparsingrules modify** to modify the parsing rule definitions. The most common use case for this command is to select one or more different templates because the certification of the customization failed.

Synopsis

```
vcb eventparsingrules modify -templateS templateName1, templateName2, …, templateNamen
-group repository -rulename rulename [-enable] [-example syslogMessage]
{ [-arg1 -arg1Value...-argN argNValue] } -user username -password password
```

Description

The **vcb eventparsingrules modify** command changes parsing rule definitions based on the templates chosen by the user. The command can also be used to add parsing rules that were inadvertently omitted when adding the parsing rule. For example, use the command to add the rules for extracting the uniqueid parameter.

Options

Table 20-14 Options and Arguments—vcb eventparsingrules modify

Option *Argument*	Description
templates *template name1, ... template namen*	Comma-separated list of event template names. Event templates are divided into categories that correspond to the function they fulfill (identification, association, and so on.) Depending upon the trap or syslog that you are adding, select no more than one template from each template category.
group *repository*	The hive under which the customizations should be made. The hive is the vendor-specific trap or syslog repository file.
rulename *ruleName*	String that is used as a key name for event rule definition.
enable	(Optional) Indicates whether the rule should be enabled or disabled. Only enabled rules are used to parse incoming traps and syslogs.
example *syslogMessage*	(Optional) Valid for syslogs only. The VCB uses this example syslog message to check the correctness of the regular expression that is automatically created by the VCB based on the user input.
variable arguments	Each event template can require different input and a different number of input parameters from none to more than one. See Event Templates Input Summary—Required and Optional Input, page 22-27.

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-15 Error Codes—vcb eventparsingrules modify

Code	Description
103	No such template name in the templates file.
211	Event template file not found.
212	Only one template can be selected from each template category.

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb eventparsingrules delete

Use the **vcb eventparsingrules delete** command to delete the parsing rule definitions of an event. Doing so does not delete or change the event template from which that event definition was cloned.

Synopsis

```
vcb eventparsingrules delete -group repository hive -rulename rulename -user username
-password password
```

Description

The **vcb eventparsingrules delete** command removes event parsing rule definitions created from an event template. It does not change or delete the event template itself.

Usage Example

```
vcb eventparsingrules delete -group cisco-syslog-repository
-rulename stack-new-master-syslog
```

This example deletes the stack-new-master-syslog rule from the cisco-syslog-repository hive.

Options

Table 20-16 Options and Arguments—vcb eventparsingrules delete

Option *Argument*	Description
group *repository hive*	The hive from which to remove the event parsing rule.
rulename *ruleName*	The rule to delete.

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-17 Error Codes—vcb eventparsingrules delete

Code	Description
222	Parsing rules repository not found.
241	No such rule name in the site.xml.

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb eventpattern Commands

Use the following **vcb eventpattern** commands to create, view, modify, and delete event patterns:

- vcb eventpattern add, page 20-28
- vcb eventpattern view, page 20-29
- vcb eventpattern modify, page 20-31
- vcb eventpattern delete, page 20-32

vcb eventpattern add

Use the **vcb eventpattern add** command to create a VNE-driver registration that points from the parsing rules hive, which is scheme or VNE-specific, to the parsing rules defined in the repository file.

Note Only those events that have this pointer are deemed as supported events. Other events are deemed generic events despite having parsing rules and event definitions.

Synopsis

```
vcb eventpattern add [-patternid patternId] -group parsing rules hive
-repository parsing rules repository hive -rulename rulename -user username
-password password
```

Description

The **vcb eventpattern add** command creates a pointer from the parsing-rules hive to the repository where the actual parsing rules are defined.

Usage Examples

```
vcb eventpattern add
-patternid 202 -group cisco-syslog-product-parsing-rules
-repository cisco-syslog-repository
-rulename stack-switch-status-syslog -user username -password password
```

Adds a pointer from the parsing rules file to the actual definitions in the parsing-rules hive with pattern ID 202. It points to the key (rule) named stack-switch-status-syslog in the cisco-syslog-repository file.

Options

Table 20-18 Options and Arguments—vcb eventpattern add

Option *Argument*	Description
patternid *patternId*	(Optional) (Recommendation: do not provide.) Unique integer to identify the supported event to VNEs. If not provided, VCB generates this number automatically. **Note** Omitting this option and argument enables the VCB to ensure that the patternid is unique and that it does not overlap with other file definitions due to registry inheritance.
group *parsing rules hive*	The hive to which this pattern should be added. The parsing-rules hives are generally scheme-specific. Device type-specific definitions can also be made.
repository *parsing rules repository hive*	The trap or syslog repository where the actual parsing rules are defined. **Note** Enter the same hive that was specified when creating parsing rules registrations using the **vcb eventparsingrules add** command.
rulename *rulename*	String that is used as a key name for event rule definition. **Note** Enter exactly the same string as the one that was specified when creating parsing rules registrations using the **vcb eventparsingrules add** command.

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-19 Error Codes—vcb event pattern add

Code	Description
221	Parsing rules hive not found.
222	Parsing rules repository not found.

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb eventpattern view

Use the **vcb eventpattern view** command to display event registrations. It is useful when you need to verify successful completion of an add command or to help find a similar case for filling in parameters on other commands.

Synopsis

```
vcb eventpattern view -group parsingrules hive -rulename { rulename | all }
[-substringmatch] [-full] -user username -password password
```

Description

The **vcb eventpattern view** command shows the actual set of events that are supported by a particular NE type or scheme. It displays the pattern ID and the repository file where the event parsing rules are defined. When the substringmatch option is used, only rules that contain a certain substring are displayed; use this option, for example, to obtain rules for a technology name such as MPLS.

Usage Examples

Example 1

```
vcb eventpattern view -group cisco-syslog-parsing-rules -rulename
stack-switch-status-syslog -user root -password admin
```

Displays the event pattern definition for the specified rulename; that is, the pattern ID, and the pattern is pointing to the parsing rules repository.

Example 2

```
vcb eventpattern view -group cisco-syslog-parsing-rules -rulename all -user root
-password admin
```

This example shows all the event pattern definitions in the specified hive.

Example 3

```
vcb eventpattern view -group cisco-syslog-parsing-rules -userdefined
-rulename bgp -substringmatch -full -user root -password admin
```

This example shows the entire event definition for all BGP events (including those defined using the VCB) defined in the cisco-syslog-parsing-rules hive. The following information is displayed:

- Pattern definitions—Parsing rules repository, pattern ID
- Parsing rules definitions—All rules in the definition that require user input, and the values set for these parameters
- Event definitions—Event attributes such as eventname, subevent names, ticketability, severity and so on

Options

Table 20-20 Options and Arguments—vcb eventpattern view

Option *Argument*	Description
group *parsingrules hive*	The parsing-rules filename used by the NE type or scheme.
rulename *rule name*	Unique string that represents the event parsing rules. **Tip** Enter **all** as the *rule name* to list all event parsing rules defined in the repository.

Table 20-20 Options and Arguments—vcb eventpattern view (continued)

Option *Argument*	Description
substringmatch	(Optional) Indicates that the rule name provided is not an exact match. This option is useful when you want to know the names of all rules that belong to a particular technology, such as BGP.
full	(Optional) Displays the entire details of the event, from the pattern definition and parsing rules to the event definition. Provides the complete picture of the how an event is supported in Prime Network. **Note** Avoid this option when using the "all" argument because it can result in a very large output.

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-21 Error Codes—vcb eventpattern view

Code	Description
221	Parsing rules hive not found.
241	No such rule name in the site.xml.

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb eventpattern modify

Use the **vcb eventpattern modify** command to modify the pointer to the parsing rules.

Synopsis

```
vcb eventpattern modify -patternid patternId -group parsing rules hive [-repository
parsing rules repository hive] [-rulename ruleName] -user username -password password
```

Description

The **vcb eventpattern modify** command modifies the pointer from the parsing-rules hive to the repository where the actual parsing rules are defined.

Usage Examples

```
vcb eventpattern modify
  -patternid 202
  -group cisco-syslog-product-parsing-rules
  -repository cisco-router-syslog-repository -user root -password admin
```

This example assumes that we are starting with the eventpattern with ID 202 that points to the cisco-syslog-repository (as shown in Usage Examples for the **vcb eventpattern add** command). In this example, we modify the repository for the eventpattern with ID 202 to the cisco-router-syslog-repository.

Options

Table 20-22 Options and Arguments—vcb eventpattern modify

Option *Argument*	Description
patternid *patternId*	Unique integer to identify the supported event to a VNE.
group *parsing rules hive*	The hive in which this pattern is to be modified. The parsing-rules hives are generally scheme-specific. VNE-specific definitions can also be made using this hive.
repository *parsing rules repository hive*	(Optional) The hive where the actual parsing rules are defined (the trap/syslog repository). Enter the same hive that was specified when creating parsing rules registrations using the **vcb eventparsingrules add** command.
rulename *ruleName*	String that is used as a key name for event rule definition. **Note** Enter exactly the same string as the one that was specified when creating parsing rules registrations using the **vcb eventparsingrules add** command.

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-23 Error Codes—vcb eventpattern modify

Code	Description
221	Parsing rules hive not found.
222	Parsing rules repository not found.
271	Pattern with ID not found

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb eventpattern delete

Use the **vcb eventpattern delete** command to delete the parsing rule from the list of supported event patterns. Doing so does not delete the parsing rules in the repository file.

Synopsis

```
vcb eventpattern delete -group parsing rule hive -patternid pattern ID -user username
-password password
```

Description

The **vcb eventpattern delete** command removes the pointer to the parsing rule defined in the repository file.

Usage Examples

```
vcb eventpattern delete -group cisco-syslog-parsing-rules -patternid 202 -user root
-password admin
```

Deletes the parsing rules pattern with ID 202. All registry entries added as a part of the **vcb eventpattern add** command will be removed from site.xml.

Options

Table 20-24 Options and Arguments—vcb eventpattern delete

Argument	Description
patternid *pattern ID*	Unique integer to identify the supported event to a VNE.
group *parsing rules hive*	The hive in which this pattern is to be modified. The parsing-rules hives are generally scheme-specific. VNE-specific definitions can also be made using this hive.

Note For the list of global options, see Global Command Options, page 18-21.

Error Codes

Table 20-25 Error Codes—vcb event pattern delete

Code	Description
221	Parsing rules hive not found.
271	Pattern with ID not found

Note For the list of general VCB error codes, see General Error Codes, page 18-22.

vcb eventarg Commands

vcb eventarg view

Use the **vcb eventarg view** command to display event parsing rule arguments and descriptions.

Synopsis

```
vcb eventarg view -user username -password password
```

Description

The **vcb eventarg view** command option displays all the VCB event parsing rules template variable arguments along with descriptions.

Troubleshooting Event Customization (CLI)

Errors that you receive from the VCB CLI are self-explanatory. Most errors make very clear what you need to do to correct the problem that has occurred. For example, if an event name already exists, you must enter a different event name. If an alarm ID or pattern ID is already in use, you should omit the related option and argument from your command and allow the VCB to generate a unique ID for you.

Note To get more information, add the **-debug** option to any **vcb** command; for more information, see Global Command Options, page 18-21.

Errors that occur in the server are not as interactive and obvious. If a newly supported event does not appear in Prime Network Events or Prime Network Vision, you need to perform some troubleshooting, as follows:

Step 1 Ensure that the device is configured to send events to Prime Network gateway. Use any tool to snoop and check whether the simulated network events that you are sending are actually arriving at the Prime Network gateway. If not, fix the issue whether it is connectivity, firewall and so on, then proceed to next step.

For detailed information, see Testing and Certifying Event Customizations, page 20-12.

Step 2 Enable debug for event processing in the VNE. Execute the following commands for the AVM that has the VNE that you will be testing and restart the AVM (not just the VNE).

```
runRegTool.sh -gs localhost set 127.0.0.1
avm<avmid>/services/logger/log4j.category.com.sheer.metrocentral.framework.eventapplicatio
n.eventcorrelation.SendAlarmMessageUtil DEBUG
runRegTool.sh -gs localhost set 127.0.0.1
avm<avmid>/services/logger/log4j.category.com.sheer.metrocentral.framework.eventmanager.Ev
entManager DEBUG
runRegTool.sh -gs localhost set 127.0.0.1
avm<avmid>/services/logger/log4j.category.com.sheer.metrocentral.framework.eventapplicatio
n.parsing.ParsingApplication DEBUG
```

Step 3 Allow the VNE to come up, then open the log file for the AVM. Check whether the newly added pattern is being loaded at VNE startup. Look for an entry in the log file that is similar to the following text:

```
DEBUG [06 21 2010 12:19:59.524 IST] - ParsingApplication.buildRulesMatrix - pattern
ATMLC-6-CLOCKING with index 5001 in the registry is now mapped into index 123 in the
parsing application.
```

The above DEBUG statement includes both the rulename (ATMLC-6-CLOCKING with) and the pattern id (5001) of the newly added event. If a similar statement is not printed for the newly added event, go to Step 4; otherwise, go to Step 5.

Step 4 Review the **vcb eventparsingrules add** command that you used, checking whether you enabled the event using the **-enable** option. If the command was issued without the **-enable** option, delete the event parsing rules using the **vcb eventparsingrules delete** command and add the event parsing rules again, ensuring that you use with the **-enable** option.

Step 5 Check statically whether the links between event pattern, event parsing rules, and event are OK. To perform this check, use the **vcb eventpattern view** command with the **-full** option (see Example 3, page 20-30). The output should display details of all the three customizations. A typo in the rulename, event type name, or event subtype name can prevent the links from being established and result in a partial display. For example, if the rulename in the **vcb eventpattern** command does not match that used in the **vcb eventparsingrules** command, only event pattern details will be displayed; details for event parsing rules and the event will not be displayed.

If the output is OK (that is, it includes details for all three customizations), go to Step 7. Otherwise, go to Step 6.

Step 6 Review the commands that have been issued and re-add or modify the customizations as required. Then go back to Step 5.

Step 7 After the static verification that you perform in step 5 succeeds, check whether the parsing itself is failing. Put a tail on the AVM log file and resend the simulated event. When parsing fails, the event is classed as a generic event. Log output similar to the following will appear.

```
DEBUG [06 21 2010 16:11:09.623 IST] - EventManager.filterEventApplications - Event
has been dropped by application
[com.sheer.metrocentral.framework.eventapplication.filter.GenericSyslogTypeFilterAp
p]

########## com.sheer.metrocentral.framework.eventapplication.types.EventData
########

  # Id            :  = 137611826381_1277116869542

  # Unique source ID:  = null

  # Type          :  = generic syslog

  # SubType       :  = generic syslog

  # SourceOID     :  = {[ManagedElement(Key=10.77.212.205)][Syslog]}

  # Event Time    :  = 1277116869542

  # Info         :  = 7.212.205 %FAN-3-FAN_0K: Fan 3 had earlier reported a rotation
error. It is ok now

  # CorrelationKeys: =

  #      CK=(MC.DA-10.77.212.205)-25:52:0:0 [16]

  # Adjacent XID  :  = null

  # Source IP interface:  = null


################################################################################
```

Step 8 Open the log file and go backwards from the end of the file until you come to the place where logs pertaining to the actual parsing process are available. Search for the string 'Testing pattern: handle *rulename*', where rulename is the string you used in the **vcb eventpattern add** command. Here you will find logs that report the results of testing each rule. Identify the rule that failed as shown in the following log.

```
DEBUG [06 21 2010 16:11:09.622 IST] - ParsingApplication.processEvent - Exception
during parsing correlation rules, at pattern-125, rule-2
Stack:[(uniqueid=>3),(syslog=>7.212.205 %FAN-3-FAN_0K: Fan 3 had earlier reported a
rotation error. It is ok now),(subtypekey=>0K),(.prulescache=>[]),(counter=>0)]
Event Data :
  ############## com.sheer.metrocentral.framework.eventapplication.parsing.types
.RawSyslogEventData #############
  # Id            :  = 4311876356_1277116869490
  # Unique source ID:  = null
  # Type          :  = raw event
  # SubType       :  = raw syslog
  # SourceOID     :  = null
  # Event Time    :  = 1277116869494
  # Info         :  = 7.212.205 %FAN-3-FAN_0K: Fan 3 had earlier reported a rotation
error. It is ok now
  # syslog = 7.212.205 %FAN-3-FAN_0K: Fan 3 had earlier reported a rotation error. It
is ok now
################################################################################
##################################
      java.lang.reflect.InvocationTargetException
        at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method) at
sun.reflect.NativeConstructorAccessorImpl.newInstance(NativeConstructorAccessorImpl
.java:39)
        at
sun.reflect.DelegatingConstructorAccessorImpl.newInstance(DelegatingConstructorAcce
ssorImpl.java:27)
        at java.lang.reflect.Constructor.newInstance(Constructor.java:513)
        at
com.sheer.metrocentral.framework.correlation.parsing.ChangeArgumentValue.execute
(ChangeArgumentValue.java:68)
```

In the above example, the parsing rule that failed is ChangeArgumentValue. The failure implies that the replacing rules that map the network event parameters to the Prime Network event subtypes are failing. Review the replacing_rules arguments used in the **vcb eventparsingrules** command and make the necessary changes.

The list of parsing rules (classes) and the corresponding option in vcb are given in the following table. Review the parameter values of the failing option and make appropriate changes.

Step 9 Restart the AVM and repeat the above steps until all errors are resolved and the event is parsed correctly and the Prime Network event is generated as expected.