Cisco Prime Network 5.3 BQL Integration Developer Guide

January 2020

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

Abstract

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON INFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Integration Developer Guide

Contents

1 Preface

This guide describes how to use the Cisco Prime Network integration interfaces. Prime Network supports the following integration interfaces:

- Prime Network Broadband Query Language (BQL)—A simple XML-based query language that provides programmatic access to the entire Prime Network information model, as well as other Prime Network key features and functions. See IMO and BQL.
- Web service—Implemented using Java API for XML Web Services (JAX-WS) framework. This provides the capability to run BQL commands over web services (WS) connections. The web service client supports execution of BQL commands and registration for notifications. It is also compatible with other WS integration tools. See Running BQL Using Web Services.
- BQL commands over HTTP interface—Provides the capability to run BQL commands via HTTP request using a standard web browser. See Running BQL using the Web Interface.
- SNMP notifications interface—Based on the Cisco EPM Notification MIB. It is used for integrations with fault management systems, such as Netcool, NetBoss, and so on. See Using Event Notification Service.
- Prime Network Shell interface— A command-line interface (CLI) for performing remote administrative tasks. See Cisco Prime Network Shell Interface.

This preface contains the following sections:

Content is not changed from Prime Network 4.2

1.1 Audience

This guide is intended for integration developers (typically professional service engineers, system integrators, or IT personnel) who want to integrate Prime Network into their Operations Support Systems (OSS). This type of integration is mostly focused on creating OSS client applications.

To support these integration developers, Cisco has exposed Prime Network functionality and its information model through the Prime Network Integration APIs, a standards-based, programmatic management interface technology.

You can use the Prime Network Integration APIs to:

- Enhance the functionalities provided by Prime Network, such as extracting network inventory, invoking service activations, or receiving alert notifications.
- Implement functionalities not provided by Prime Network, such as activation, troubleshooting, and specialized reporting.
- Integrate Prime Network with third-party OSS applications, such as inventory management and fault management systems.

1.2 Document Organization

This document includes the following topics.

Table 1-1 Organization

Topic Title	Description
Overview	Introduces the Prime Network integration interfaces. This topic includes the section Introducing Cisco Prime Network Integration.
IMO and BQL	Describes the Prime Network Information Model Objects (IMO) framework, BQL, and BQL commands. This topic includes: • Understanding IMO • • Note: For more information on other possible Media type to poll inventory information through NBI, see Media Types to Poll Inventory Information • Understanding BQL • Managing Faults using BQL • BQL Commands
BQL Implementation	Describes the three different execution modes for BQL. This topic includes: • Running BQL Using Secured Socket Communication • Running BQL Using Web Services • Running BQL using the Web Interface
BQL Application Examples	Explains the BQL notification service and provides BQL examples for Prime Network applications. Also, provides the best practices for BQL parsing. This topic includes:

<table>
<tr><th>Topic Title</th><th>Description</th></tr>
<tr><td></td><td>

Best Practices for BQL Parsing
Processing BQL Notification Messages
Administering Cisco Prime Network Using BQL
Retrieving Inventory Data Using BQL
Generating Standard Reports Using BQL
Running Command Builder Scripts Using BQL
Managing Soft Properties Using BQL
BQL Error Catalog and Examples

</td></tr>
<tr><td>Event Notification Service</td><td>Describes the Prime Network Event Notification Service and provides the details on the CISCO-EPM-NOTIFICATION-MIB mapping within Prime Network. This section includes:

Using Event Notification Service
Understanding the Cisco EPM Notification MIB
Event Notification Service Errors and Exceptions
Sample SNMP Notification Examples

</td></tr>
<tr><td>Cisco Prime Network Shell Interface</td><td>Describes the Prime Network Shell (CLI) interface. This section includes:

Introducing the Cisco Prime Network Shell Interface
Understanding Cisco Prime Network Shell Behavior
Cisco Prime Network Shell Errors

</td></tr>
<tr><td>Appendixes</td><td>Lists the BQL commands for the corresponding Prime Network GUI tasks. Also, describes the productivity tools that are useful in identifying the BQL commands and the OID values and types using Prime Network GUI. This section includes:

Appendix A, “Cisco Prime Network GUI and BQL Mapping”
Media Types to Poll Inventory Information

This section describes media type integer mapping details to poll the inventory information from Prime through NBI.
<table>
<tr><th>Media Type</th><th>Integer Mapping Value</th><th>Description</th></tr>
</table>

</td></tr>
</table>

Topic Title	Description

COAX	1	Coax
Thin_COAX	2	Thin coax
Thick_COAX	3	Thick coax
Fiber_OPTIC	4	Fiber Optic
MULTIMODE_FO	5	Multi mode fiber optic
SINGLEMODE_FO	6	Single mode fiber optic
SHORT_SINGLEMODE_FO	7	Short single mode fiber optic
LONG_SINGLEMODE_FO	8	Long single mode fiber optic
UTP	9	UTP
STP	10	STP
FTP	11	FTP
EIA_TIA_232	12	
EIA_TIA_449	13	
V_35	14	
X_21	15	
EIA_TIA_530	16	
EIA_TIA_530A	17	
GENERIC_SERIAL	18	
EIA_TIA_612_613	19	

- Appendix B, “Productivity Tools”

1.3 Conventions

This document uses the following conventions:

Convention	Indication
bold font	Commands and keywords and user-entered text appear in bold font.
italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in *italic* font.

Convention	Indication
[ ]	Elements in square brackets are optional.
{x | y | z }	Required alternative keywords are grouped in braces and separated by vertical bars.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
String	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
courier font	Terminal sessions and information the system displays appear in `courier` font.
< >	Nonprinting characters such as passwords are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Note Means reader take note. Notes contain helpful suggestions or references to material not covered in the publication.

1.4 Additional User Documentation

For a complete set of Prime Network 4.2.2 documentation, see the *Cisco Prime Network Documentation Overview, 4.2.2*.

1.5 Obtaining Documentation, Obtaining Support, and Security Guidelines

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:
http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html
Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS version 2.0.

2 Overview

This section introduces the various aspects of Prime Network integration.

2.1 Introducing Cisco Prime Network Integration

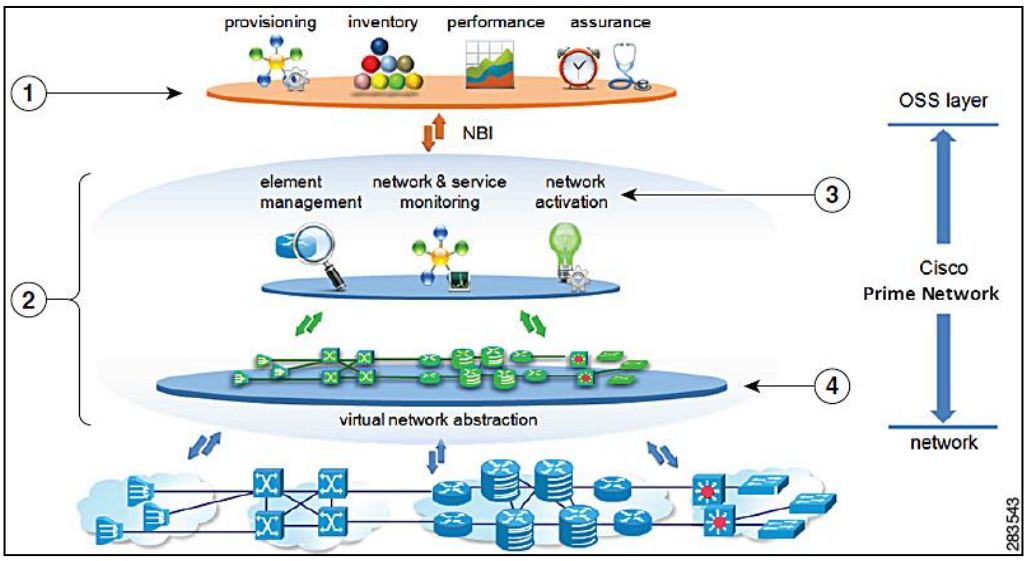
Cisco Prime Network is a network management foundation for Cisco-based service provider networks. It addresses the challenge of managing diverse converged, multitechnology, multilayer, and multivendor IP next-generation networks (IP NGNs). Prime Network supports operators of all Service Provider IP NGN networks, particularly:

- Converged IP/MPLS core and service edge networks.
- IP Radio Access Network (RAN) backhaul (Mobile Transport over Packet [MTOP]) networks.
- Carrier Ethernet networks.

A key Prime Network objective is to assist network operators with trouble resolution. Prime Network's ability to discover and represent relationships and associations among network features facilitates quick and accurate isolation of problems and verification of expected feature configurations.

Prime Network’s unique model-based virtual network abstraction serves as a live information foundation, presenting the operator with a consistent and complete end-to-end topological view of network resources, technologies, and services. Prime Network simplifies integration tasks with a variety of operational support system (OSS) applications, including provisioning, resource inventory management, performance management, and service assurance (see Figure 2-1).

Figure 2-1 Prime Network Architecture



1	OSS Application	**3**	Prime Network applications
2	Prime Network	**4**	VNE Layer

OSS integration developers have access to an extensive Northbound Interface (NBI) SDK, exposing the network abstraction model and other APIs, as well as a complete set of developer support resources and Cisco's advanced technical services. Prime Network provides the tools and foundation for an integrated suite of OSS capabilities.
The *Cisco Prime Network 4.2.2 Customization User Guide* describes the various extensions and customization options that extend the Prime Network information model, such as creating configuration scripts, workflow sequences, soft properties, threshold crossing alarms, and business tags.

This guide focuses on the programmatic integration of Prime Network with other OSS systems. These integrations generally fall into the following categories:

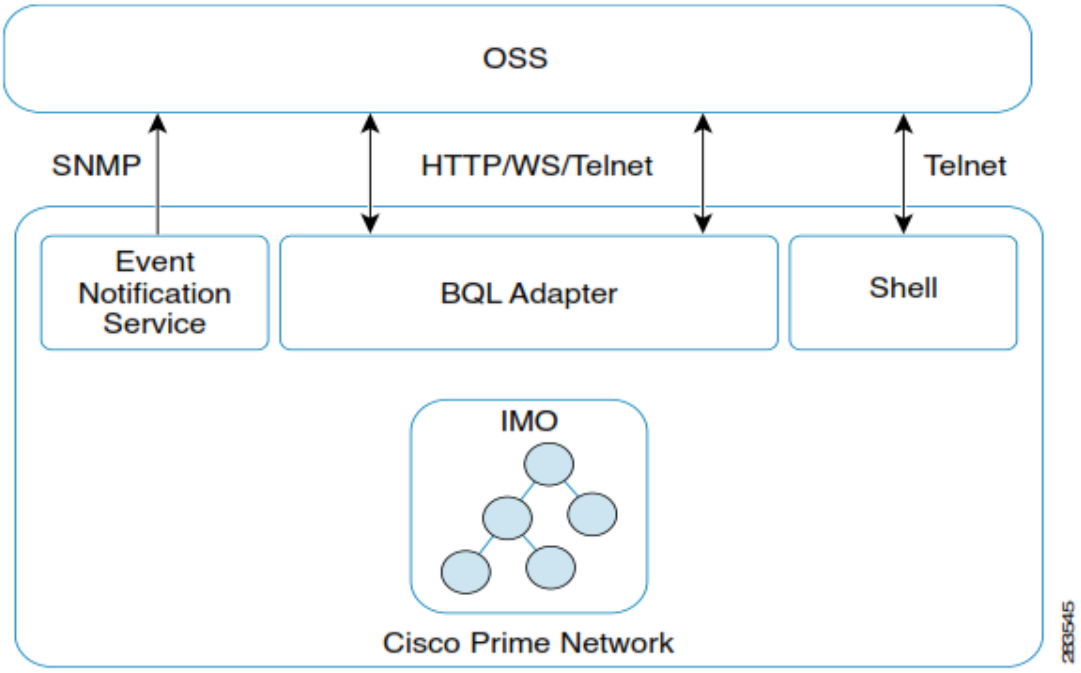
- Inventory retrieval for inventory management.
- Integration with Prime Network fault-related notifications for network assurance.
- Execution of configuration command scripts and workflows for activation and diagnostics.
- Remote Prime Network administration.

Prime Network provides a number of different NBI integration methods, giving the OSS developer a choice. Its main integration interface is called Broadband Query Language (BQL), an XML-based query language, which can be used over a Telnet-like connection, or as a web service, over HTTP(S). BQL uses an XML representation of the Prime Network information model, in the form of Information Model Objects (IMOs).

Figure 2-2 shows the following interfaces that are supported in Prime Network:

- Event Notification Service—Fault-related notifications.
- BQL Adapter—Inventory (IMO) retrieval, full fault management, command execution, and Prime Network administration.
- Prime Network Shell—Prime Network administration through a command-line interface (CLI).

Figure 2-2 Prime Network Interfaces



Event Notification Service

Prime Network supports a notification service that informs northbound fault systems of faults and other events in Prime Network and the managed network elements (NEs). An OSS client application can configure and subscribe to this Event Notification Service (ENS) to receive notification on actionable, non-actionable, and Element Management System (EMS)-generated internal events.

Prime Network sends these notifications over the SNMP protocol (v1 or v2) in a format that conforms to the standard Cisco EPM Notification MIB (CISCO-EPM-NOTIFICATION-MIB).

See Using Event Notification Service.

BQL Adapter

BQL is a simple XML-based query language that provides programmatic access to the entire Prime Network information model, as well as other Prime Network key features and functions. It uses IMO, which is based on the TeleManagement Forum TMF-513/608 standards.
There are several ways to use the BQL interface. The simplest is to open a Telnet or SSL connection to the Prime Network gateway, and communicate directly. Alternatively, BQL commands can be plain commands sent over HTTP to the Prime Network embedded web server, or Prime Network can support BQL communication as a web service.
See
Note: For more information on other possible Media type to poll inventory information through NBI, see Media Types to Poll Inventory Information

Understanding BQL.

Prime Network Shell

Cisco Prime Shell Interface is a deprecated interface which is not maintained in Prime Network. This interface will **not** work on a fresh installation of Prime Network.
When upgrading from an older version that supported the Shell Interface to new Prime Network release, the Shell Interface will continue to work as before.

The Prime Network shell interface is a CLI for performing remote administration tasks. It supports a subset of the administration management of Prime Network, including the configuration of AVMs and Virtual Network Elements (VNEs), remote execution of command scripts, and so on. All of these tasks are also supported over BQL, the preferred administrative interface.
For more information, see Cisco Prime Network Shell Interface.

2.1.1 Cisco Developer Network Support for Prime Network

The Prime Network Technology Center is an online resource for integration developers who use Prime Network APIs. It provides information, guidance, and examples to help you integrate your applications with Prime Network. It provides you a platform to interact with Prime Network subject matter experts.

The Prime Network Technology Center includes resources, such as:

- BQL examples, which help you retrieve inventory data, manage Prime Network administration, manage the network faults, and configure network elements.
- BQL libraries, which help you integrate with Java clients.
- Command Builder scripts, which help you manage faults, configuration, and performance for DWDM and SONET controllers, Session Border Controllers, and so on.

- *Cisco Prime Network Information Model* Javadoc.
- Prime Network *ISDK* Java application samples.

You can access the Prime Network Technology Center website using this URL: *https://developer.cisco.com/site/prime-network/.*

You must have a Cisco.com account with partner level access, or you must be a Prime Network licensee to view some of the information on the Prime Network Technology Center website.

To know more about Cisco Developer Network membership and programs, visit: *http://developer.cisco.com/web/partner/join.*

2.1.2 Advanced Services

To get the highest value from Prime Network, it must be installed and configured with the final operations processes in mind. Cisco's Advanced Services group offers a broad array of services to help ensure that each Prime Network deployment is as fast and smooth as possible, optimizing the benefits of Prime Network. From initial process evaluations to specifying the most effective system configuration, integration, and implementation, Cisco Advanced Services is ready to provide customized assistance. For more information about Cisco Advanced Services for Prime Network, contact your local Cisco account team or send an e-mail to wwsp-onm-bus-dev@cisco.com.

3 IMO and BQL

This topic contains the following sections:

- Understanding IMO
-
- Note: For more information on other possible Media type to poll inventory information through NBI, see Media Types to Poll Inventory Information

- Understanding BQL
- Managing Faults using BQL
- BQL Commands

3.1 Understanding IMO

The Cisco Prime Network Information Model Objects (IMO) framework is a standard for generic information representation. It defines the representation of multiple-vendor, multiple-technology, and multiple-layer network and service information, based on TMF-513/608 Multi-Technology Network Management (MTNM) recommendations. Internally, within Prime Network, IMO is implemented as a set of interface classes for representing all network and service information objects. However, for external integration purposes, the IMO specification defines 1:1 bidirectional translation of all IMOs to XML. External applications that access Prime Network via BQL are exposed only to the XML formatting of the Prime Network information.

IMO is only a temporary *packaging* mechanism. The live network data model maintained in Prime Network is stored within the VNEs in the form of Device Component (DC) hierarchies, while persistent information (such as events, alarm history, or service entities) is kept in the Prime Network database. When interfacing with external systems, Prime Network works with transient IMOs, which translate to and from the actual data elements in the system. This enables sending information from and to Prime Network in a generic, consistent way (analogous to SNMP, in which the MIB is not the actual information repository, but only a data reference map, interacting with client applications through transient protocol data units [PDUs]).

Figure 3-1 shows the context of the IMO within Prime Network.

Figure 3-1 Prime Network IMO

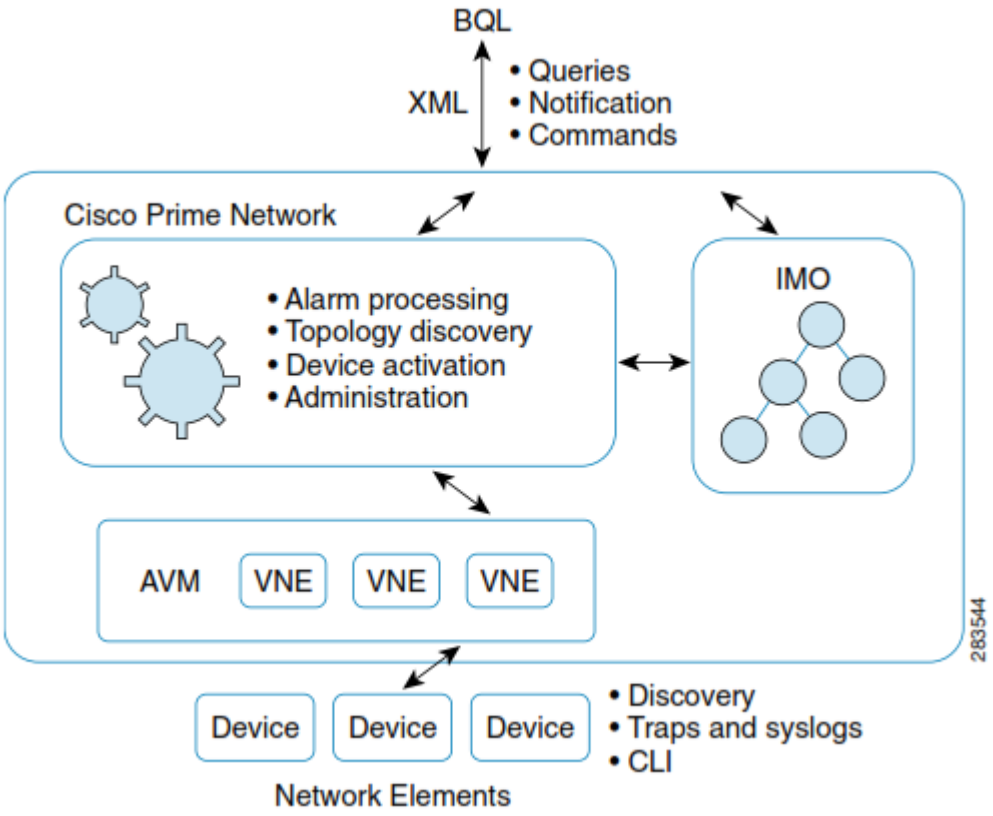


Table 3-1 presents the general XML format of an IMO.

Table 3-1 XML format of an IMO

Element	Details
IMO	<IMO_TYPE> IMO_OID IMO_PROPERTIES </IMO_TYPE>
IMO_OID	<ID type="Oid"> OID_FORMAT </ID>
IMO_PROPERTIES	SIMPLE_PROPERTY | PROPERTIES_ARRAY
SIMPLE_PROPERTY	<PROPERTY_NAME type=PROPERTY_TYPE> PROPERTY_VALUE </PROPERTY_NAME>
PROPERTIES_ARRAY	<IMObjects_Array> IMO 1 IMO 2 ... </IMObjects_Array>
OID_FORMAT	A valid string encoding of an OID.
PROPERTY_NAME	String representing a property name.

Element	Details
PROPERTY_TYPE	Value can be a: • Primitive type; for example, int, short, long, char, and so on. • Basic java types; for example, String, Integer, Long, and so on. • Prime Network complex types; for example, IPAddress, IPSubnet, and so on. • Prime Network enumerations; i.e. any class inherit from EnumerationBase for example EnabledStateEnum. • Any other IMO or OID type. **Notes**: 1. For detailed information on the complex types, see the package com.sheer.types.* in the Cisco Prime Network Information Model Objects Javadoc. You can access this document from the Prime Network Technology Center website. 2. For detailed information on the enumerations, see the package com.sheer.types.enum in the Cisco Prime Network Information Model Objects Javadoc. You can access this document from the Prime Network Technology Center website.
PROPERTY_VALUE	String representing a property value.

For details on the way information objects are represented and structured in Prime Network, see the *Cisco Prime Network Information Model Objects* Javadoc. You can access this document from the Prime Network Technology Center website.

3.1.1 IMO OIDs

An IMO identifier (OID) is the unique identifier of every IMO instance in the system (similar to OIDs in SNMP MIBs). The OID uniquely identifies every IMO by providing a cascading structure that describes the location of the entity. For example, the OID of a specific port in a typical NE is formatted by cascading the NE name, shelf number, module number, and port.

Different device types can have different OID schemas. For example, a port OID in an NE might also include such items as submodules or subslots.

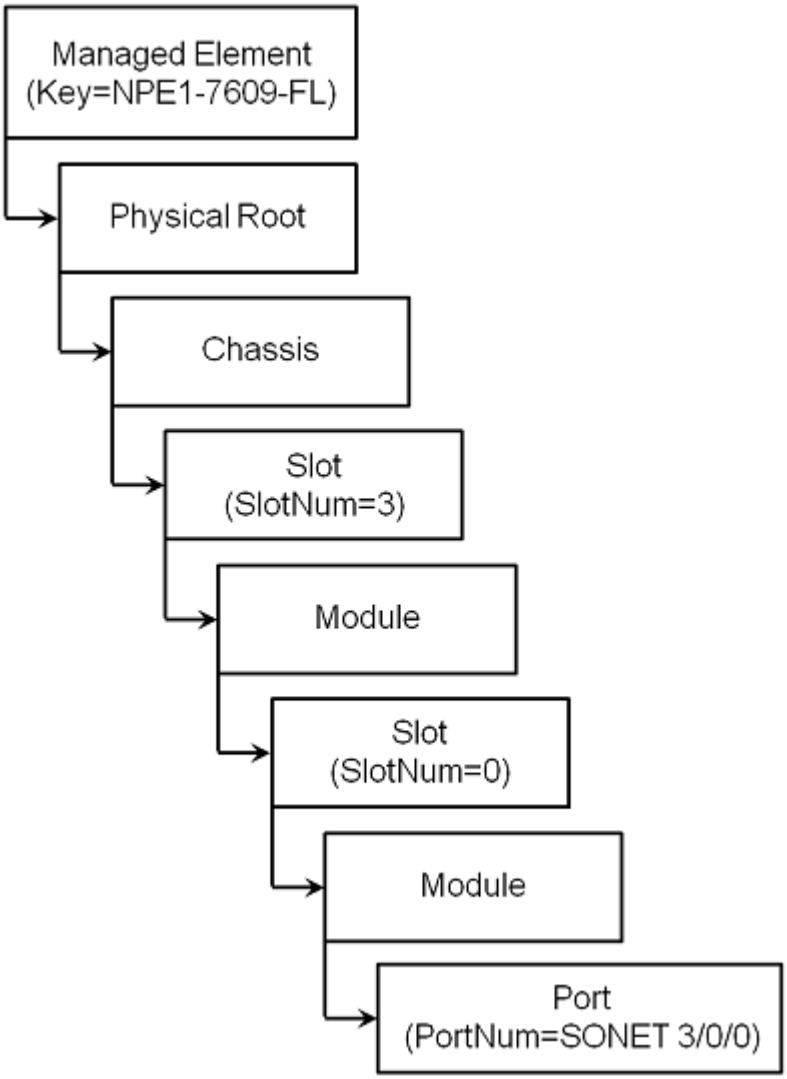
3.1.1.1 IMO OID Example

In this example, the OID of port SONET 3/0/0.
The port 0 which resides on a sub module located in slot 0, of a module which resides in slot 3 of the NE identified as NPE1-7609-FL is:

```
{[ManagedElement(Key=NPE1-7609-
FL)][PhysicalRoot][Chassis][Slot(SlotNum=3)][Module][Slot(SlotNum=0)][Modu
le][Port(PortNumber=SONET 3/0/0)]}
```

Figure 3-2 illustrates the hierarchy of this OID.

Figure 3-2 OID Hierarchy



3.1.1.2 OID Interface Mapping

In Prime Network, the OID format of an interface or a port in generic but in most of the cases will contain the interface name which helps you to identify the mapping to the interface / port.

Example 1: OID of a physical port FastEthernet4/1

```
cenAlarmManagedObjectClass(.13) (Event Source OID):
{[ManagedElement(Key=AGG1)][PhysicalRoot][Chassis][Slot(SlotNum=4)][Module]
[Port(PortNumber=FastEthernet4/1)][PhysicalLayer][Syslog]}
```

Example 2: OID of IP interface GigabitEthernet1/0.100 under VRF cust1

```
cenAlarmManagedObjectClass(.13) (Event Source OID):
{[ManagedElement(Key=PE1)][LogicalRoot]
[FWComponentContainer(Type=3)][Vrf(VrfName=cust1)][IpInterface(IpInterfaceN
ame=GigabitEthernet1/0.100)]}
```

In some cases different devices will have different naming for the same type of interface, and the interface name might not be included as part of the OID. However, you can use the last part of the OID to identify the interface based on OS type (Cisco IOS or Cisco IOS XR.)

Example 3: Different mapping of LAG interface OID

```
cenAlarmManagedObjectClass(.13) (Event Source OID):
{[ManagedElement(Key=msn1)][LogicalRoot][Context(ContextName=Default
context)][DataLinkAggregationContainer(Type=1)]
[DataLinkAggregation(Index=2)][Syslog]}
```

DataLinkAggregationContainer is a container for all the Link Aggregation (LAG) and Ethernet channels. The name of such an interface is:

- Port-channel in Cisco IOS
- Bundle-ether in Cisco IOS XR

In the OID example, you can see a DataLinkAggregation with index value 2. Thus, the interface mapping will be to **Port-channel2** (in Cisco IOS) or **Bundle-ether2** (in Cisco IOS XR).

Example 4: Mapping based on interface type (MLPPP)

```
cenAlarmManagedObjectClass(.13) (Event Source OID):
{[ManagedElement(Key=msn1)][LogicalRoot][Context(ContextName=Default
context)][EncapsulationAggregationContainer(Type=1)]
[EncapsulationAggregation(Group=4)][Syslog]}
```

EncapsulationAggregationContainer is a container for Multilink Point-to-Point Protocol (MLPPP). The name of such interfaces in Cisco IOS or Cisco IOS XR is **Multilink**. In the OID example, you can see an EncapsulationAggregation with index value 4. Thus, the interface mapped is **Multilink4**.

3.1.2 IMO Properties

IMO is the data schema of the network. All network and service information is modeled in IMO. Each IMO has properties, which contain the actual data. The properties can be of several types:

- Primitive type; for example, int, short, long, char, and so on.
- Basic java types; for example, String, Integer, Long, and so on.
- Prime Network complex types; for example, IPAddress, IPSubnet, and so on.
- Prime Network enumerations; i.e. any class inherit from EnumerationBase for example EnabledStateEnum.
- Any other IMO or OID type.

Notes:

1. For detailed information on the complex types, see the package *com.sheer.types.** in the *Cisco Prime Network Information Model Objects Javadoc*. You can access this document from the Prime Network Technology Center website.
2. For detailed information on the enumerations, see the package *com.sheer.types.enum* in the *Cisco Prime Network Information Model Objects Javadoc*. You can access this document from the Prime Network Technology Center website.

The properties can appear as single scalars (for example, the IP address of an NE) or as part of an array (for example, a list of physical ports in a card).

The IMOs contain references to other, related IMOs. These relations describe network dependencies:

- Containment—Cards within a Chassis or ports within a card.
- Logical reference—A BFD session or OSPF interface configuration associated with an IP interface.
- Connectivity—Physical connection between ports, association between port to LAG interface and a reference of forwarding entry to an interface.

3.1.3 IMO Concepts

The following sections describe the various IMO concepts:

3.1.3.1 Constructs

The interobject references enable Prime Network to pack and provide IMO data in the form of object constructs. A construct is a set (hierarchy) of interrelated objects. Every IMO construct has a *root object*, which is the entry point to the objects collection, through which the construct can be traversed. The root object is the starting point of the BQL query.

The object constructs can contain any graph of objects (supporting references in any direction), and not just "top-down" hierarchical trees. For example, an IMO construct whose root object is a specific NE card might contain a reference to the chassis in which it resides and which is its ancestor in the physical network hierarchy.

IMO constructs support recursive nesting of IMOs as well as arrays of IMOs. IMO constructs can nest an IMO within another IMO. Several IMOs can be aggregated into an IMO array; both can be combined to create complex IMO constructs.

The containment (IMO nesting) represents physical or logical relations between the objects; for example:

- Equipment containment (cards within a chassis, ports within a card)
- Connectivity (physical connectivity, interfaces grouping)
- Logical association (logical configuration applied to interface)

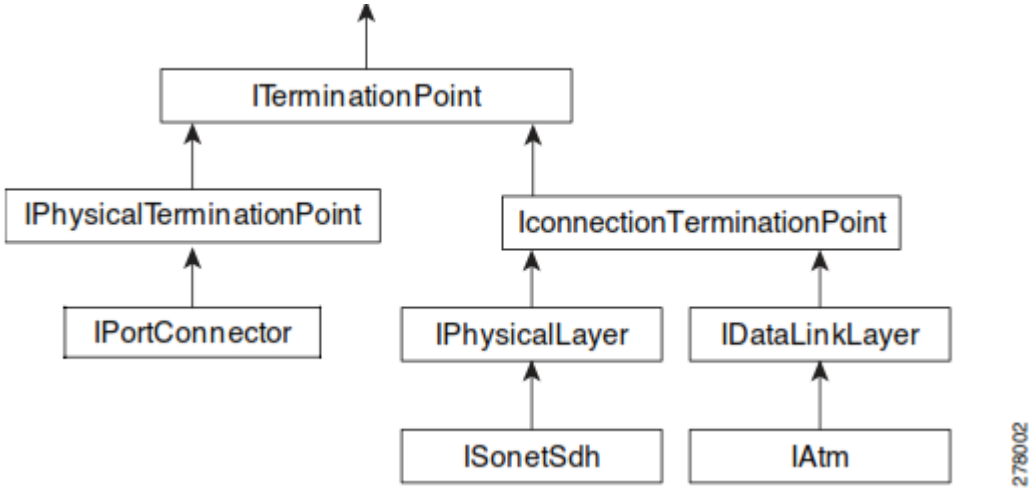
Each of the properties of an IMO can be an IMO by itself. In this case, the type of the property is one of the existing IMO types.

3.1.3.2 Inheritance

The IMO interface types are defined through an object oriented inheritance scheme. All IMO types are defined in a common inheritance tree, with the common root (base-class) type IMO.

For example, CTP objects, such as IAtm or ISonetSdh, are all derived from (are children of) a common base type, IConnectionTerminationPoint.

Figure 3-3 IMO Inheritance – an example



3.1.3.3 Aspect

IMOs can contain *aspect*. The aspect mechanism enables extending IMOs by nesting other objects as extended properties. Aspect is similar to the **JOIN** command in Structured Query Language (SQL). This mechanism enables extending IMOs (representing network resources) that have additional information from the Prime Network database (alarms, business information). The extension is non-persistent (lives only throughout the scope of the BQL **Get** command).

For example, an IMO representing a port can be extended with an aspect that contains subscriber information. Another example is an IMO construct of a NE with its associated alarms; the IMO representing the NE is extended with aspects, which contain the associated alarms.

You can attach aspect to every IMO by specifying the required aspect in the requiredAspects and excludedAspects sections in the retrieval specification. See **Format and Syntax of Aspect and Retrieval Specification**, on page 14 for details.

For examples on how an aspect and retrieval specification is used in the BQL **Get** command, see:

- Retrieval Specification, page 14
- Sample BQL Scripts for Managing Faults, page 26

3.1.3.4 Retrieval Specification

The retrieval specification (RS) defines the scope of information to be retrieved by a **Get** command. It describes the IMOs that are returned as well as the properties that are returned in each IMO. The RS allows you to include or exclude every element in the result.

The RS also allows registering for change notifications to be sent whenever a property changes within the specified data scope. Registration for changes enables, for example, receiving notifications on port or module status change, or any changes in the configuration of the network.

You can define properties that should be included (requiredProperties) or excluded (excludedProperties) for each of the IMO types. However, the RS does not support filtering of object instance by value. For more details, see Format and Syntax of Aspect and Retrieval Specification, page 14.

For examples on how RS is used in the BQL **Get** command, see:

- Sample BQL Scripts for Managing AVMs and VNEs, page 115
- Sample BQL Scripts for Retrieving Inventory Data, page 132
- Sample BQL Scripts for Managing Faults, page 26

Format and Syntax of Aspect and Retrieval Specification

The XML layout of the Aspect and RS is:

```
<param name="rs">
<value>
<key name="[rs-name]">
  <entry name="register">[true/false]</entry>
  <entry name="emptyInitialImo">[true/false]</entry>
  <entry name="emptyNotifications">[true/false]</entry>
  <key name="requiredProperties">
      <key name=[* or IMO type]>
        <entry name=[* or property name]/>
        <entry name=[* or property name]/>
      . . .
      </key>
    </key>
    <key name="excludedProperties">
      <key name=[* or IMO type]>
        <entry name=[* or property name]/>
        . . .
      </key>
  </key>
  <key name="requiredAspects">
    <key name=[* or oid type]>
      <entry name=[* or aspect oid type]/>
      <entry name=[* or aspect oid type]/>
      . . .
    </key>
  </key>
```

```
  <key name="excludedAspects">
   <key name=[* or oid type]>
       <entry name=[* or aspect oid type]/>
       . . .
   </key>
   </key>
</key>
</value>
</param>
```

This example demonstrates the following conventions:

- The name of the RS (rs-name) is optional and used only for readability of the XML file.
- The register entry indicates whether to register for changes on the objects that match the RS. This flag is optional and can be omitted. If it is not specified, it is assumed to be false.
- The aspect is used only to attach the database OID to the network IMO. The actual data content of the database IMO construct is specified in the requiredProperties section (the same as for the network IMOs).
  For example: The requiredAspects section extends the ManagedElement IMO (the NE) to include the database IMO of type IAlarmListOid (which is the OID of the alarm container). However, to retrieve the actual alarm information, it must be specified in the requiredProperties section. The excludedAspects section is used only to fine-tune the attachment definition, and not to specify the retrieval data scope.
- The requiredProperties key indicates properties to include in the result. The excludedProperties key describes properties to exclude from the result. The requiredAspects key indicates aspects to attach to the result. These modifiers are optional and can be discarded from the RS.
- For included and excluded properties, you can specify a property name or wildcard (*) to represent all properties.
- A property that is both required and excluded is excluded.
- emptyInitialImo and emptyNotifications parameters are supported by Get command if the IMO is IManagedElement type or a tree under it. emptyInitialImo defines whether you can receive the current information about the IMO; emptyNotifications defines whether the change notifications received after running the Get command includes the change or just the change notification.
  Note: The emptyInitialImo and emptyNotifications parameters could be set to true only if the “register” parameter is set to true.

3.1.3.5 Dereferencing

Each IMO has a unique ID called Object ID (OID). When one IMO serves as a property for another IMO, there might be no direct object and property relationship between them, but instead a reference relation using the OID. For example, instead of an IMO instance

serving as a property for another, there might be an OID instance representing the IMO instance. In this case, to retrieve the data, two queries are required: One to get the first IMO with the OID of the referenced IMO, and another to get referenced IMO by its OID. Getting the referenced IMO by its OID in a single query using aspect is called Dereferencing.

For example, when you execute the **Get** command on IVirtualRouter, observing its retrieved properties, you can see that it contains a property called Vrf. It is expected that the value of the property is an IVrf IMO, but this property is actually IVrfOid; that is, the ID of the IVrf IMO. To retrieve the IVrf data, you need to execute another **Get** command using the IVrfOid to retrieve the IVrf data. Dereferencing enables you to retrieve the IVirtualRouter data, including the referenced IVrf in one query, by retrieving the IVrf data as an aspect of IVirtualRouter.

3.1.4 IMO Example

This section provides a simple example of IMOs and constructs. It describes their content structure as well as their XML encoding.

Consider the following IMO:

```
<IPhysicalLayer>
   <ID type="Oid">{[ManagedElement(Key=ana-dev-7609-
2)][PhysicalRoot][Chassis][Slot(SlotNum=3)][M
odule][Slot(SlotNum=0)][Module][Port(PortNumber=GigabitEthernet3/0/0)][PhysicalLay
er]}</ID>
   <AdminStatusEnum type="Integer">1</AdminStatusEnum>
   <LastChanged type="java.util.Date">Sun Sep 13 18:17:58 PDT 2009</LastChanged>
   <MaxSpeed type="com.sheer.types.Speed">1000.0 Mbps</MaxSpeed>
   <MediaTypeEnum type="Integer">4</MediaTypeEnum>
   <OperStatusEnum type="Integer">1</OperStatusEnum>
   <TypeEnum type="Integer">6</TypeEnum>
   <mtu type="Long">1500</mtu>
   <portDescription type="String">Link to 7609-1</portDescription>
</IPhysicalLayer>
```

The IMO type is IPhysicalLayer and represents the physical properties of an Ethernet port.

Table 3-2 summarizes the properties contained in this IMO.

Table 3-2 Example IMO Properties

Property	Type	Value	Description
ID	Oid	{[ManagedElement(Key=ana-dev-7609-2)][PhysicalRoot][Chassis] [Slot(SlotNum=3)]	GigabitEthernet3/0/0 port in subslot 0 of slot 3 in the ana-dev-7609-2.

Property	Type	Value	Description
		[Module][Slot(SlotNum=0)][Module] [Port(PortNumber=GigabitEthernet3/0/0)][PhysicalLayer]}	
AdminStatusEnum	Integer	1	Up.
OperStatusEnum	Integer	1	Up.
mtu	Long	1500	Maximum transmission unit.
LastChanged	java.util.Date	Sun Sep 13 18:17:58 PDT 2009	Last operational change date.
MaxSpeed	com.sheer.types.Speed	1000.0 Mb/s	A Cisco basic type that contains the speed in the unit (Mb/s).
MediaTypeEnum	Integer	4	Fiber Optic
TypeEnum	Integer	6	IANA IfType for Ethernet – ethernetCsmacd

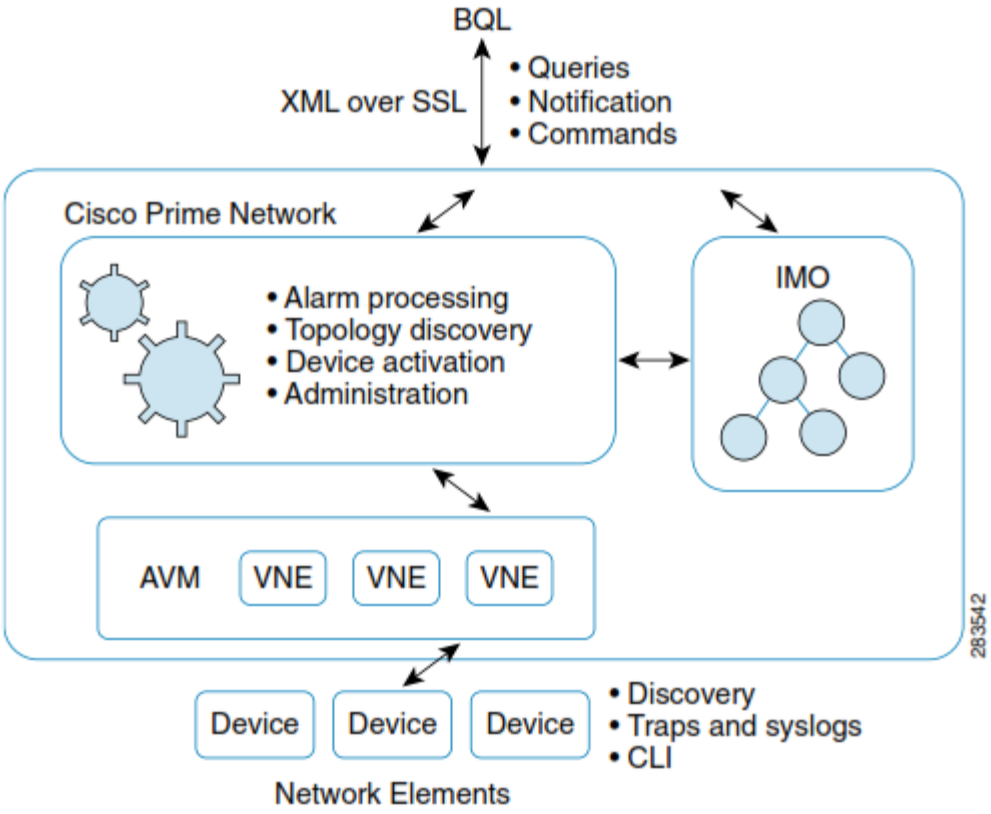
Note: For more information on other possible Media type to poll inventory information through NBI, see Media Types to Poll Inventory Information

3.2 Understanding BQL

Broadband Query Language (BQL) is a simple XML-based query language that provides programmatic access to the entire Cisco Prime Network information model, as well as other Prime Network key features and functions.

Figure 3-4 shows an overview of the BQL and how it relates to other functions of Prime Network.

Figure 3-4 An Overview of BQL



Queries, Notifications, and Commands

BQL is a query language used for retrieving Prime Network data (similar to how SQL is used for querying data from a relational database). BQL contains commands for:

- Performing queries and modifying Prime Network data.
- Registering for future notifications whenever there is a change in Prime Network data.
- Sending commands to the network elements managed by Prime Network.

IMO

An Information Model Object (IMO) is a packaging mechanism for passing information between Prime Network gateway and client applications, including the BQL. The applications that access Prime Network via BQL are exposed to IMO as an XML representation of the Prime Network information model.

IMO is based on the TMF-513/608 MTNM. For more details on IMO, see Understanding IMO, page 6.

VNE Network Integration

Prime Network maintains an abstraction of each network element that it manages. This abstraction is called Virtual Network Element (VNE). The VNE is created when Prime Network discovers the network element, and it is updated periodically with new information when Prime Network polls the network element or when a new notification

(trap or syslog) is received from the network element. The VNEs are the source of information about the topology and status of the managed NEs.

Prime Network utilizes the VNE to collect information from a single NE. The VNE polls the NE by sending queries through standard management interfaces such as SNMP, XML, CLI (Telnet / SSH) and HTTP. The remote network elements and the VNE network abstraction layer are sources that can be accessed via BQL.

3.2.1 Connecting to the BQL Adapter

All data exchanged between the BQL adapter (part of the Prime Network gateway) and clients is formatted as XML messages, containing data objects. The BQL adapter (part of the Prime Network gateway) and remote clients communicate in one of the following interaction schemes:

- TCP sockets—Provides the capabilities to run BQL commands over TCP socket on port 9002 or 9003 (SSL port) and establishes a BQL session. See Running BQL Using Secured Socket Communication.
- Web services—Implemented using Java API for XML Web Services (JAX-WS) framework. This provides the capability to run BQL commands over a WS connection. See Running BQL Using Web Services.
- BQL commands over HTTP interface—provides the capability to run BQL commands via HTTP request using a standard web browser. See Running BQL using the Web Interface.

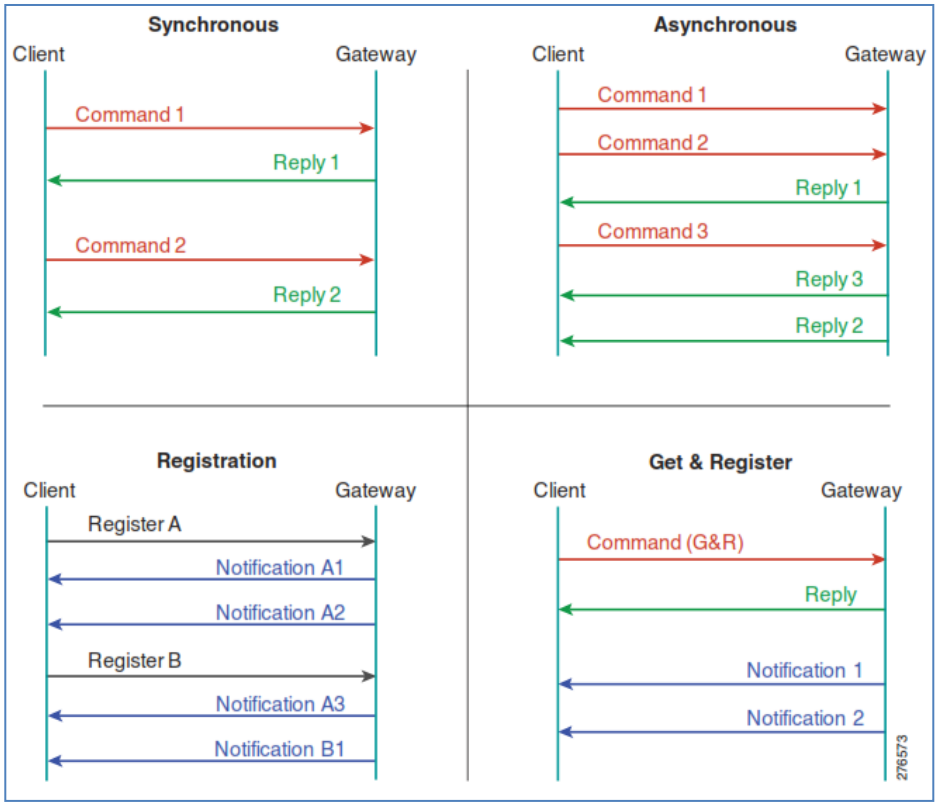
3.2.2 BQL Interaction Modes

BQL supports the following interaction modes with OSS applications (see Figure 3-5):

- Synchronous requests (also known as blocking mode)
- Asynchronous requests (also known as nonblocking mode)
- Register for notifications, using Get command or Register command.
- Get and Register for notifications

Note If your OSS application uses Asynchronous mode, it may be difficult for your application to bind a response with a command request, especially while running commands on different NEs. BQL supports addition of a command ID, which you can associate with the notification message for a specific command request. See Processing BQL Notification Messages, page 87 to understand the syntax for the command ID.

Figure 3-5 BQL Interaction Modes



BQL Commands

A set of BQL basic commands consists of:

- Get—Retrieve an information query. In case the retrieval specification includes VNEs only, you can use Get for register for change notifications, or for register for notifications which does not include the change.
- Register—Register for change notifications.
- Create—Create a new data object.
- Update—Set properties and relations in existing objects.
- Delete—Delete an existing object.
- Refresh—Poll data on demand.

For more details, see BQL Commands, page 47.

Get Command

Get is a commonly used BQL command. The Get command behaves according to the parameters given in the retrieval specification. The Get command retrieves an XML data construct (IMO construct) for a single object or an array or hierarchy of objects; for example, an NE with all its cards and ports, or an end-to-end service path. The results of a query can be filtered based upon the OID of the root IMO construct and certain properties; for example, type of IMOs, set of properties for each specific IMO, and values defined in Retrieval Specification. The information about other related objects can be

included in the results (defined in an Aspect). The Get command also enables registering for notifications on any future changes in the returned objects using a register option. For more details, see BQL Commands, page 47.

Registrations and Notifications

As a result of the Get or Register BQL commands, the interface receives notifications that indicate an object has been changed, added, or deleted. A notification is an object that describes the change in another object. It is an array in which each element represents a single property change.

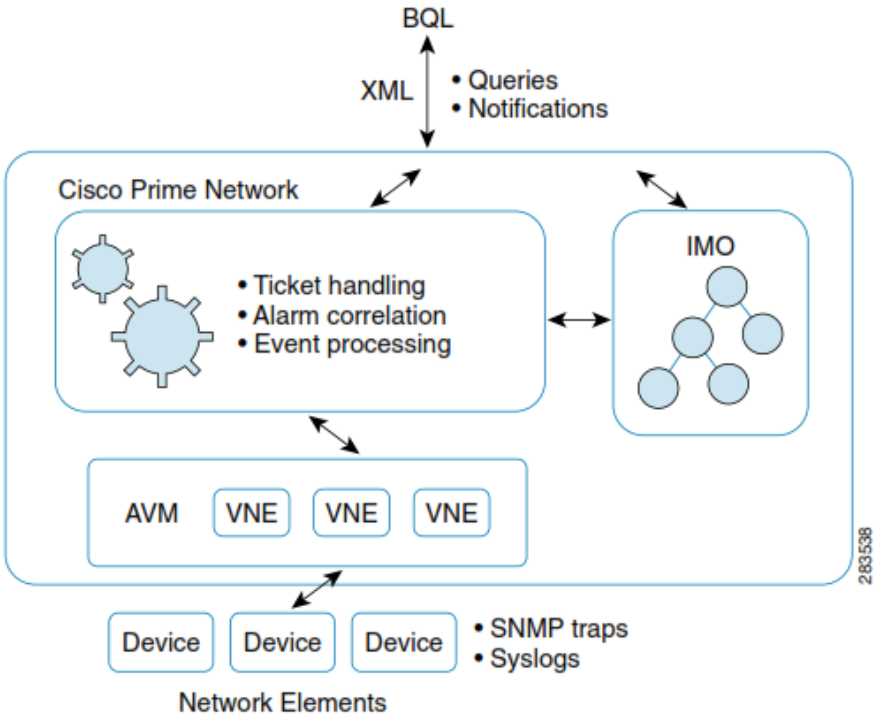
For more details, see BQL Commands, page 47 and Processing BQL Notification Messages, page 87.

3.3 Managing Faults using BQL

Cisco Prime Network analyzes and manages faults by implementing event collection, identification, and correlation functionality. After identifying the event, Prime Network groups related events and uses the automatically discovered virtual network model to perform fault inspection and advanced correlation to determine the root cause of the fault and create a ticket.

Figure 3-6 shows a high-level overview of how BQL is used to manage faults.

Figure 3-6 Managing Faults using BQL



Using Prime Network fault management queries, the client application can:

- Retrieve tickets, alarms, and events.

- Subscribe to notifications of changes (add, modify, or delete) to tickets, alarms, and events using the **Get** and **Register** commands. These notification commands are used to subscribe for an existing event ID.
- Subscribe to notification of changes for a particular event using the **RegisterEventNotifications** command. This notification command is used to subscribe for a new event based on event category and severity.
- Perform the following tasks on the tickets:
  - Acknowledge
  - Clear
  - Remove
  - Clear and remove

See the *Cisco Prime Network 4.2.2 User Guide* for more information on managing tickets.

Additional Reading

- Review *Cisco Prime Network 4.2.2 Administrator Guide* to understand the Prime Network user roles and scopes.
- Review *Cisco Prime Network 4.2.2 User Guide* to understand the fault management implementation in Prime Network.
- Review *Cisco Prime Network 4.2.2 Supported Cisco VNEs* to learn about supported NEs, and *Cisco Prime Network 4.2.2 Supported Service Alarms* and *Cisco Prime Network 4.2.2 Supported Traps* and supported alarms and traps for the NEs.
- See the *Cisco Prime Network Information Model* Javadoc to understand the IMO for VNE, events, alarms, and tickets. This document is available on the Prime Network Technology Center website. You must have a Cisco.com account with partner level access, or you must be a Prime Network licensee to access this website.

Note

- For information about IMO and fault model changes after upgrading to Prime Network 4.2.2, see the Cisco Prime Network 4.2.2 Installation Guide.
- For information about the ticket and alarms unique type changes in Prime Network 4.2.2, see the Cisco Prime Network 4.2.2 Installation Guide. You might need to update the existing BQL scripts after upgrading to Prime Network 4.2.2.
- Also, refer Using Event Notification Service, page 353 for information on receiving notifications due to network element events (traps and syslog messages).

Fault Management Terms

The following table defines some of the Prime Network fault management terms that are used in this section and in Using Event Notification Service, page 353.

Table 3-3 Fault Management Terms

Term	Definition
actionable events	Subset of EC-recognized events that are of interest to the Prime Network fault management subsystem. Actionable events cause Prime Network to take an action such as deduplication, correlation, impact analysis, and so on. The Prime Network fault management subsystem must have a parser defined for actionable events.
EC	Prime Network Event Collector. Component used in Prime Network to collect, filter, store, and forward raw event notifications from the network. EC is the ingress point into the Prime Network system for event notifications.
EMS internal event	Event generated by Element Management System (EMS) (also referred to as active monitoring); for example, “AVM 777 is shutting down. Unit = 10.56.22.25” or “AVM 333 - OutOfMemory - Unit 10.56.58.180.”. In Prime Network, system event, security event, provisioning event, and audit event are considered to be EMS internal events.
Event Notification Service	Prime Network-supported notification service that generates EPM traps and supports sending actionable and non-actionable event notifications, tickets, ticket updates and EMS-generated internal events.
EventID	ID used for EMS internal events; for example, system event, security event, provisioning event, and audit event.
network event	Event that is related to the network element (NE). Network events could be SNMP traps or syslogs generated by the NE (also referred to as passive monitoring). Events can be differentiated into actionable and nonactionable.
NetworkEventID	ID used for network events.
Nonactionable events	Syslogs and traps for which corresponding VNEs do not have a parser. Starting from 4.2.2 Prime Network does a best effort at extracting information from these syslogs and traps, and save them in the database as archived. No additional actions are taken on these events (no association, correlation, impact analysis and so on). The term nonactionable events and standard events are synonymous
Raw Event Notification Service	A notification service supported by Prime Network that maps incoming event notifications (for example, syslogs and traps)

Term	Definition
	into a normalized trap format (EPM-NOTIFICATION-MIB) and emits trap notifications.

3.3.1 Registering for Notification for Specific Events

Using the RegisterEventNotifications command you can register for notification for a particular event. This command can be used for the following event categories:

- Actionable events
- EMS internal event (System, Security, Audit, and Provisioning)

This command is implemented similar to a search operation; where the search string (events that need to be filtered) is provided as an array of IMOs. Each event in the Prime Network database is compared with the search string and a notification is generated and forwarded to the client.

Retrieval specification and IMO inheritance are not supported and should not be used while using this command.

The following filter properties are supported:

- Event type; for example, Link down, Port up, login authentication failed syslog, and so on.
- SeverityEnum. For detailed information on the enumerations, see the package *com.sheer.types.enum* in the *Cisco Prime Network Information Model Objects Javadoc*. You can access this document from the Prime Network Technology Center website.

The BQL notification does not include the Provisioning event descriptions and the nonactionable events.

The UnsupportedOperationException is displayed when the unsupported filter properties are defined.

Note The RegisterEventNotifications command may cause performance deterioration. We recommend that only a fewer event filters are defined based on your requirements.

The notification message is sent as `IObjectCreateNotification`. See Sample Notification for a Security Event and Sample Notification for V2 Traps.

Unregistering from Notifications

After registering to one or more notifications, a client application should unregister when the notifications are no longer needed. You can unregister from notifications by closing the socket connection. This implicitly unregisters all registrations.

3.3.2 Fault Management Interfaces

Table 3-4 lists the BQL commands supported for fault management queries.

Table 3-4 Supported Fault Management BQL Command Queries

Command Name	IMO/OID Type	IMO/OID Value	Description
Get	com.sheer.imo.keys.ITicketListAspectOid	{[TicketListAspect]}	Retrieves all active tickets. It does not retrieve events that are correlated to each ticket.
Get	com.sheer.imo.keys.INewAlarmOid	{[NewAlarm(id=<AlarmID>]}	Retrieves details of a specific alarm using AlarmId (the ID of an alarm).
Get	com.sheer.imo.keys.IEventOi	{[Event(Id=<EventId>)]}	Retrieves details of EMS internal events (Audit, Provisioning, System, and Security) using EventId (the ID of the event).
Get	com.sheer.imo.keys.INetworkEventOid	{[NetworkEvent(Id=<NetworkEventID>)]}	Retrieves details of a specific network event (syslog, trap, and Service alarm) using NetworkEventId (the ID of the network event).

Table 3-5 lists the BQL commands supported for fault management operations.

Table 3-5 Supported Fault Management BQL Command Operations

Command Name	IMO/OID Type	IMO/OID Value	Description
Acknowledge	com.sheer.imo.keys.INewAlarmOid	{[NewAlarm(Id=<TicketID>)]}	Acknowledges a ticket.
ForceClear	com.sheer.imo.keys.INewAlarmOid	{[NewAlarm(Id=<TicketID>)]}	Approves the reported faulty ticket and clears the faulty networking entity from Prime Network.
Remove	com.sheer.imo.keys.INewAlarmOid	{[NewAlarm(Id=<TicketID>)]}	Removes the ticket and all its active sub tickets.

AddNote	com.sheer.imo.keys.IEventOid	{[Event(Id=<Event Id>)]}	Adds note for the selected ticket.
RegisterEvent Notifications	—	—	Register for notifications based on the event type and SeverityEnum filters.
EnableIdentified UnmanagedElements	com.sheer.imo.keys.IIdentifiedUnmanagedElements	{[IdentifiedUnmanagedElements]}	Enables / disables the support of unmanaged devices.
GetIdentified UnmanagedElements	com.sheer.imo.keys.IIdentifiedUnmanagedElements	{[IdentifiedUnmanagedElements]}	Gets the list of the supported unmanaged subnets.
SetIdentified UnmanagedElements	com.sheer.imo.keys.IIdentifiedUnmanagedElements	{[IdentifiedUnmanagedElements]}	Sets a list of supported unmanaged subnets.

See Processing BQL Notification Messages, page 87 to understand how to register for notification messages whenever correlated alerts are reported by Prime Network or a trouble ticket client application needs to open tickets.

3.3.3 Sample BQL Scripts for Managing Faults

This section contains the following sample BQL scripts:

The Mediator Debugger tool helps you identify the BQL commands for any Prime Network GUI task (except for Cisco Prime Network Events tasks). Using this tool, you can write your own BQL commands for the required GUI tasks. You cannot register for notification using Cisco Prime Network GUI.

Enable / Disable the Support of Unmanaged Devices

The following example shows the usage of the BQL **EnableIdentifiedUnmanagedElements** command to enable or disable the support of unmanaged devices.

```
<command name="EnableIdentifiedUnmanagedElements">
  <param name="oid">
        <value>{[IdentifiedUnmanagedElements]}</value>
  </param>
  <param name="SupportEnabled">
        <value>true</value>
  </param>
</command>
```

Get the list of the Supported Unmanaged Subnets

The following example shows the usage of the BQL **GetIdentifiedUnmanagedElements** command to get the list of the supported unmanaged subnets.

```
<command name="GetIdentifiedUnmanagedElements">
  <param name="oid">
        <value>{[IdentifiedUnmanagedElements]}</value>
  </param>
</command>
```

Set a list of Supported Unmanaged Subnets

The following example shows the usage of the BQL **SetIdentifiedUnmanagedElements** command to get the list of the supported unmanaged subnets.

```
<command name="SetIdentifiedUnmanagedElements">
  <param name="imobject">
```

```
      <value>
        <IIdentifiedUnmanagedElements>
              <ID type="Oid">{[IdentifiedUnmanagedElements]}</ID>
        <SupportEnabled type="Boolean">true</SupportEnabled>
            <IdentifiedUnmanagedIPSubnets
        type="com.sheer.types.IPSubnet_Array">
                  <com.sheer.types.IPSubnet>0.0.0.0,0.0.0.0</com.sheer.ty
            pes.IPSubnet>
            </IdentifiedUnmanagedIPSubnets>
        </IIdentifiedUnmanagedElements>
      </value>
  </param>
</command>
```

Getting Details of All Tickets

The following example shows the usage of the BQL **Get** command to retrieve all active tickets in Prime Network. It does not retrieve events that are correlated to each ticket.

```
<command name="Get">
<param name="oid">
<value>{[TicketListAspect]}</value>
</param>
<param name="rs">
<value>
<key name="">
  <entry name="depth">100</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.newalarm.ITicketListAspect">
              <entry name="Tickets" />
        </key>
  <key name="com.sheer.imo.newalarm.ITicket">
        <entry name="*"/>
      </key>
  </key>
</key>
</value>
</param>
</command>
```

Getting Details of a Ticket

The following example shows the usage of the BQL **Get** command to retrieve details of a specific ticket (ticket ID is 212).

```
<command name="Get">
<param name="oid">
<value>{[NewAlarm(Id=212)]}</value>
</param>
<param name="rs">
<value>
<key name="">
```

```
  <entry name="depth">10</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
     <key name="com.sheer.imo.newalarm.ITicket">
        <entry name="*"/>
     </key>
  </key>
</key>
</value>
</param>
</command>
.
```

Getting Details of a Ticket with the Correlation Alarms Details

The following example shows the usage of the BQL **Get** command to retrieve all details of a specific ticket (ticket ID is 212). Here,

Retrieval Specification (rs) is used to get the correlation alarm details.

```
<command name="Get">
  <param name="oid">
        <value>{[NewAlarm(Id=212)]}</value>
  </param>
  <param name="rs">
        <value><key name="com.sheer.imo.keys.INewAlarmOid">
  <entry name="depth">100</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">true</entry>
  <key name="requiredProperties">
      <key name="com.sheer.imo.newalarm.IAlarm">
        <entry name="Source"/>
        <entry name="CausedAlarms"/>
        <entry name="CausingAlarmSource"/>
        <entry name="LatestState"/>
        <entry name="LastEventTime"/>
        <entry name="SeverityEnum"/>
        <entry name="TicketOid"/>
        <entry name="AutoCleared"/>
        <entry name="Archived"/>
        <entry name="EventSequence"/>
        <entry name="AckStateEnum"/>
    </key>
    <key name="com.sheer.imo.newalarm.ITicket">
        <entry name="CausedAlarms"/>
        <entry name="AutoCleared"/>
        <entry name="EventSequence"/>
        <entry name="Archived"/>
        <entry name="Note"/>
        <entry name="AggregatedSeverityEnum"/>
        <entry name="AckStateEnum"/>
        <entry name="DuplicationCount"/>
        <entry name="AffectedDevicesCount"/>
        <entry name="Source"/>
        <entry name="CausingAlarmSource"/>
        <entry name="EventCount"/>
        <entry name="LatestState"/>
        <entry name="AggregatedAckStateEnum"/>
        <entry name="LastEventTime"/>
        <entry name="SeverityEnum"/>
        <entry name="TicketOid"/>
        <entry name="LastModificationTime"/>
        <entry name="ReductionCount"/>
        <entry name="AlarmCount"/>
  </key>
  <key name="com.sheer.imo.newalarm.INetworkEvent">
        <entry name="*"/>
  </key>
</key>
<key name="requiredAspects">
  <key name="com.sheer.imo.keys.IOid">
        <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
  </key>
  <key name="com.sheer.imo.keys.INewAlarmOid">
        <entry name="com.sheer.imo.keys.IAlarmBusinessObjectOid"/>
```

```
  </key>
</key>
</key></value>
  </param>
</command>
.
```

Getting Ticket Details for All Managed Network Elements

The following example shows the usage of the BQL **Get** command to retrieve all active tickets for the managed network elements in Prime Network. It does not retrieve events that are correlated to each ticket.

```
<command name="Get">
<param name="oid">
<value>{[ManagedElementSnapshot]}</value>
</param>
<param name="rs">
<value>
  <key name="Alarms per NEs">
      <entry name="register">false</entry>
      <key name="requiredProperties">
      <key name="com.sheer.imo.topology.IManagedElementSnapshot">
      <entry name="*"/>
  </key>
  <key name="com.sheer.imo.IManagedElement">
      <entry name="IP"/>
    <entry name="DeviceName"/>
    <entry name="Alarms"/>
 </key>
 <key name="com.sheer.imo.newalarm.ITicketListAspect">
       <entry name="*"/>
  </key>
  </key>
  <key name="requiredAspects">
       <key name="com.sheer.imo.keys.IManagedElementOid">
       <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
  </key>
  </key>
  </key>
</value>
</param>
</command>
.
```

Getting Ticket Details for a Managed Network Element

The following example shows the usage of the BQL **Get** command to retrieve all active tickets for a managed network element (CiscoGSRXR) in Prime Network. It does not retrieve events that are correlated to each ticket.

```
<command name="Get">
<param name="oid">
<value>{[ManagedElement(Key=CiscoGSRXR)]}</value>
</param>
<param name="rs">
<value>
```

```
<key name="Alarms per NEs">
<entry name="register">false</entry>
<key name="requiredProperties">
<key name="com.sheer.imo.topology.IManagedElementSnapshot">
<entry name="*"/>
</key>
<key name="com.sheer.imo.IManagedElement">
<entry name="IP"/>
<entry name="DeviceName"/>
<entry name="Alarms"/>
</key>
<key name="com.sheer.imo.newalarm.ITicketListAspect">
<entry name="*"/>
</key>
</key>
<key name="requiredAspects">
<key name="com.sheer.imo.keys.IManagedElementOid">
<entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
</key>
</key>
</key>
</value>
</param>
</command>
.
```

Getting a Ticket List of Network Element from Physical Inventory

The following example shows the usage of the BQL **Get** command to retrieve all active tickets, alarms, and events from the port (Ethernet0/1) for a managed device (P-North) in Prime Network.

```
<command name="Get">
<param name="oid">
<value>{[ManagedElement(Key=P-
North)][PhysicalRoot][Chassis][Slot(SlotNum=0)][Module][Port
(PortNumber=Ethernet0/1)][PhysicalLayer]}</value>
</param>
<param name="rs">
<value><key name="PortConnectorController">
      <entry name="depth">10</entry>
      <entry name="register">true</entry>
      <entry name="cachedResultAcceptable">false</entry>
    <key name="requiredAspects">
        <key name="com.sheer.imo.keys.INEOid">
          <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
        </key>
    </key>
    <key name="requiredProperties">
        <key name="ITicketListAspect">
              <entry name="*"/>
        </key>
        <key name="ITicket">
              <entry name="*"/>
        </key>
        <key name="IAlarm">
              <entry name="*"/>
```

```
        </key>
        <key name="INetworkEvent">
              <entry name="*"/>
        </key>
      </key>

</key></value>
</param>
</command>
.
```

Getting a Ticket List of Network Element from Logical Inventory

The following example shows the usage of the BQL **Get** command to retrieve alarms for a managed device in Prime Network (P-West-IOU-154). Here, the service alarm is being retrieved from LSE OID.

```
<command name="Get">
<param name="oid">
<value>{[ManagedElement(Key=P-West-IOU-
154)][LogicalRoot][FWComponentContainer(Type=4)][Ls
e][ServiceEvent(DiffObject=169.254.154.213:0)][TicketList]}</value>
</param>
<param name="rs">
<value>
<key name="">
  <entry name="depth">1</entry>
  <entry name="register">false</entry>
  <entry name="cachedResultAcceptable">false</entry>
<key name="requiredAspects">
        <key name="com.sheer.imo.keys.INEOid">
              <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
        </key>
</key>
<key name="requiredProperties">
  <key name="com.sheer.imo.newalarm.ITicket">
      <entry name="LastModificationTime"/>
      <entry name="EventCount"/>
      <entry name="Visible"/>
      <entry name="SuppressDisplay"/>
      <entry name="Source"/>
      <entry name="Name"/>
      <entry name="AckStateEnum"/>
      <entry name="AggregatedSeverityEnum"/>
      <entry name="LastEventTime"/>
      <entry name="Archived"/>
      <entry name="AutoCleared"/>
      <entry name="EventSequence"/>
      <entry name="CausingAlarmSource"/>
      <entry name="CausingAlarmName"/>
      <entry name="TicketOid"/>
    </key>
  </key>
</key>
</value>
</param>
</command>
```

.

Getting Details of a Network Event

The following example shows the usage of the BQL **Get** command to retrieve details of a network event (1430398437889_1260302002869). The NetworkEventId is ID for syslogs, traps, or Service alarms.

```
<command name="Get">
  <param name="oid">
        <value>{[NetworkEvent(Id=1430398437889_1260302002869)]}</value>
  </param>
  <param name="rs">
        <value><key name="com.sheer.imo.keys.INetworkEventOid">
  <entry name="depth">100</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">true</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.newalarm.INetworkEvent">
              <entry name="*"/>
        </key>
  </key>
</key></value>
  </param>
</command>
```

Getting Details of an EMS Event

The following example shows the usage of the BQL **Get** command to retrieve details of an EMS event (4928). The EventId is ID for Audit, Provisioning, System, and Security events.

```
<command name="Get">
  <param name="oid">
        <value>{[Event(Id=4928)]}</value>
  </param>
  <param name="rs">
        <value><key name="com.sheer.imo.keys.IEventOid">
  <entry name="depth">100</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">true</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.newalarm.IEvent">
              <entry name="*"/>
        </key>
  </key>
</key></value>
  </param>
</command>
```

.

Getting Affected Parties

The following example shows the usage of the BQL **Get** command to retrieve details of affected parties by the ticket (ticket ID 6).

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Get">
  <param name="oid">
        <value>{[NewAlarm(Id=6)]}</value>
  </param>
  <param name="rs">
      <value><key name="get-affected-snc">
  <entry name="depth">100</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.IMO">
              <entry name="*"/>
        </key>
  </key>
  <key name="excludedProperties">
        <key name="com.sheer.imo.ITrapValue">
              <entry name="*"/>
        </key>
  </key>
  <key name="requiredAspects">
        <key name="com.sheer.imo.keys.IOid">
              <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
        </key>
  <key name="com.sheer.imo.keys.INewAlarmOid">
        <entry name="com.sheer.imo.keys.IAlarmBusinessObjectOid"/>
  </key>
  <key name="com.sheer.imo.keys.IAffectedSNCOid">
        <entry name="com.sheer.imo.keys.IAffectedPartiesOid"/>
  </key>
</key>
</key></value>
  </param>
</command>
.
```

Adding a Note to a Ticket

The following example shows the usage of the BQL **AddNote** command to add notes for a selected ticket.

```
<command name="AddNote">
  <param name="oid">
        <value>{[NewAlarm(Id=6)]}</value>
  </param>
  <param name="note">
        <value>Add the note text here</value>
  </param>
</command>
.
```

The following example shows the usage of the BQL **AddNote** command with an external username (for example, Netcool user).

```
<?xml version="1.0" encoding="UTF-8"?>
userInfo=MyNetcoolUserName
<command name="AddNote">
  <param name="oid">
        <value>{[NewAlarm(Id=290012)]}</value>
  </param>
  <param name="note">
        <value>My Note</value>
  </param>
</command>
.
```

Acknowledging a Ticket

The following example shows the usage of the BQL **Acknowledge** command to acknowledge a ticket.

```
<command name="Acknowledge">
  <param name="oids">
        <value>{[NewAlarm(Id=1)]}</value>
  </param>
</command>
.
```

The following example shows the usage of the BQL **Acknowledge** command with an external username.

```
<?xml version="1.0" encoding="UTF-8"?>
userInfo=ExternalUserName
<command name="Acknowledge">
  <param name="oids">
        <value>{[NewAlarm(Id=290012)]}</value>
  </param>
</command>
.
```

Clearing a Ticket (Force Clear)

The following example shows the usage of the BQL **ForceClear** command to approve a reported faulty ticket and clear the faulty networking entity from Prime Network.

```
<command name="ForceClear">
  <param name="oids">
        <value>{[NewAlarm(Id=1)]}</value>
  </param>
</command>
.
```

The following example shows the usage of the BQL **ForceClear** command with an external username.

```
<?xml version="1.0" encoding="UTF-8"?>
userInfo=ExternalUserName
<command name="ForceClear">
  <param name="oids">
        <value>{[NewAlarm(Id=290012)]}</value>
  </param>
</command>
.
```

Removing a Ticket (Clear and Remove)

The following example shows the usage of the BQL **Remove** command to remove a ticket and all its active sub tickets.

```
<command name="Remove">
  <param name="oids">
      <value>{[NewAlarm(Id=5)]}</value>
```

```
      <value>{[NewAlarm(Id=6)]}</value>
  </param>
</command>
.
```

The following example shows the usage of the BQL **Remove** command with an external username (e.g. Netcool user.)

```
<?xml version="1.0" encoding="UTF-8"?>
userInfo=ExternalUserName
<command name="Remove">
  <param name="oids">
        <value>{[NewAlarm(Id=290012)]}</value>
  </param>
</command>
.
```

Registering for Notification for all SNMPv1 Traps

The following example shows the usage of the BQL **RegisterEventNotifications** command to register for notification for all SNMPv1 traps.

```
<command name="RegisterEventNotifications">
  <param name="imobjectArr">
        <value>
              <newalarm.IV1TrapEvent/>
        </value>
  </param>
  <param name="rs">
      <value>
        <key name="">
              <entry name="register">true</entry>
        </key>
      </value>
  </param>
</command>
.
```

To register notification for SNMPv2 and SNMPv3 traps, change newalarm.IV1TrapEvent to newalarm.IV2TrapEvent and newalarm.IV3TrapEvent for SNMPv2 and SNMPv3 traps respectively.

Registering for Notification for a Specific Security Event Type

The following example shows the usage of the BQL **RegisterEventNotifications** command to register for notification for a security event type 404 (login).

```
<command name="RegisterEventNotifications">
  <param name="imobjectArr">
        <value>
              <newalarm.ISecurityEvent>
                    <Name type="Integer">404</Name>
              </newalarm.ISecurityEvent>
        </value>
  </param>
  <param name="rs"<
```

```
        <value>
              <key name="">
                    <entry name="register">true</entry>
              </key>
        </value>
  </param>
</command>
.
```

In the above example, to register for other event types change newalarm.ISecurityEvent to

- INetworkEvent for all Syslog and Service events (alarms).
- IProvisioningEvent for all Provisioning events.
- ISystemEvent for all System events.
- IAuditEvent for all Audit events.

Registering for Notification for Syslog from a Specific NE

The following example shows the usage of the BQL **RegisterEventNotifications** command to register for notification for syslog from an NE 169.254.207.60_3.

```
<command name="RegisterEventNotifications">
  <param name="imobjectArr">
        <value>
          <newalarm.INetworkEvent>
              <Source type="Oid">
              {[ManagedElement(Key=169.254.207.60_3)][Syslog]}</Source>
           </newalarm.INetworkEvent>
        </value>
  </param>
  <param name="rs">
        <value>
              <key name="">
                    <entry name="depth">20</entry>
                    <entry name="register">true</entry>
              </key>
        </value>
  </param>
</command>
.
```

Registering for Notification for Security Event Based on Severity

The following example shows the usage of the BQL **RegisterEventNotifications** command to register for notification for a Security event with severity CLEARED (2).

```
<?xml version="1.0" encoding="UTF-8"?>
  <command name="RegisterEventNotifications">
        <param name="imobjectArr">
              <value>
                  <newalarm.ISecurityEvent>
                    <SeverityEnum type="Integer">2</SeverityEnum>
                  </newalarm.ISecurityEvent>
```

```
            </value>
        </param>
        <param name="rs">
            <value>
                <key name="">
                    <entry name="register">true</entry>
                </key>
            </value>
        </param>
  </command>
```

.

Registering for Notification for Multiple Events

The following example shows the usage of the BQL **RegisterEventNotifications** command to register for notification for a Security event with event type (404) and V1 traps with severity 6 (critical).

```
<command name="RegisterEventNotifications">
  <param name="imobjectArr">
        <value>
            <newalarm.ISecurityEvent>
                <Name type="Integer">404</Name>
            </newalarm.ISecurityEvent>
          <newalarm.IV1TrapEvent>
            <SeverityEnum type="Integer">6</SeverityEnum>
          </newalarm.IV1TrapEvent>
        </value>
  </param>
  <param name="rs">
        <value>
            <key name="">
                <entry name="register">true</entry>
            </key>
        </value>
  </param>
</command>
```

.

Sample Notification for a Security Event

Registering for a Security Event

The following example shows the usage of the BQL **RegisterEventNotifications** command to register for notification for a Security event type 404 (login).

```
<?xml version="1.0" encoding="UTF-8"?>
      <command name="RegisterEventNotifications">
            <param name="imobjectArr">
                  <value>
                        <newalarm.ISecurityEvent>
                              <Name type="Integer">404</Name>
                        </newalarm.ISecurityEvent>
                  </value>
            </param>
          <param name="rs">
            <value>
```

```
                  <key name="">
                        <entry name="register">true</entry>
                  </key>
            </value>
          </param>
</command>
```

Notification for a Security Event

The following example shows the notification `IObjectCreateNotification` for a Security event type 404 (event ID 8026).

```
<?xml version="1.0" encoding="UTF-8"?>
<IMObjects_Array>
      <IObjectCreateNotification type="IObjectCreateNotification"
      instance_id="0">
            <ID type="Oid">{[Notification]}</ID>
            <Imo type="newalarm.ISecurityEvent" instance_id="1">
                  <ID type="Oid">{[Event(Id=8026)]}</ID>
                  <ClientTypeEnum type="Integer">0</ClientTypeEnum>
                  <Description type="String">Successful login
                  root</Description>
                  <DetectionTime type="java.util.Date">Thu Jul 29
                  17:26:25 IDT 2010</DetectionTime>
                  <Name type="Integer">404</Name>
                  <OriginatingIP
                  type="com.sheer.types.IPAddress">10.21.148.107</Origi
                  natingIP>
                  <SeverityEnum type="Integer">2</SeverityEnum>
                  <Source
                  type="Oid">{[MCNetwork][MCVM(IP=10.56.58.176)][Avm(Avm
                  Number=11)]}</Source>
                  <UserName type="String">root</UserName>
            </Imo>
      </IObjectCreateNotification>
</IMObjects_Array>
```

Sample Notification for V2 Traps

Registering for v2 Traps

The following example shows the usage of the BQL **RegisterEventNotifications** command to register for notification for V2 traps.

```
<?xml version="1.0" encoding="UTF-8"?>
      <command name="RegisterEventNotifications">
            <param name="imobjectArr">
                  <value>
                        <newalarm.IV2TrapEvent />
                  </value>
            </param>
            <param name="rs">
                  <value>
                        <key name="">
                              <entry name="register">true</entry>
                        </key>
                  </value>
```

```
            </param>
        </command>
```

.

Notification for V2 Traps

The following example shows the notification **IObjectCreateNotification** for a V2 trap (1025054261479_1280738820852).

```
<IMObjects_Array>
     <IObjectCreateNotification type="IObjectCreateNotification"
   instance_id="0">
     <ID type="Oid">{[Notification]}</ID>
     <Imo type="newalarm.IV2TrapEvent" instance_id="1">
         <ID type="Oid"> {[NetworkEvent (Id=1025054261479_1280738820852)]}
         </ID>
         <Archived type="Boolean">true</Archived>
         <Community type="String">public</Community>
         <Description
type="String">.iso.org.dod.internet.mgmt.mib-
2.system.sysUpTime.sysUpTimeInstance--&gt;159 days, 0 hours, 56 minutes,
28 seconds.
.iso.org.dod.internet.snmpV2.snmpModules.snmpMIB.snmpMIBObjects.snmpTrap.sn
mpTrapOID.0--&g t;.1.3.6.1.4.1.9.9.43.2.0.1
.iso.org.dod.internet.private.enterprises.cisco.ciscoMgmt.ciscoConfigManMIB
.ciscoConfigMan
MIBObjects.ccmHistory.ccmHistoryEventTable.ccmHistoryEventEntry.ccmHistoryE
ventCommandSour ce.6090--&gt;1
.iso.org.dod.internet.private.enterprises.cisco.ciscoMgmt.ciscoConfigManMIB
.ciscoConfigMan
MIBObjects.ccmHistory.ccmHistoryEventTable.ccmHistoryEventEntry.ccmHistoryE
ventConfigSourc e.6090--&gt;2
.iso.org.dod.internet.private.enterprises.cisco.ciscoMgmt.ciscoConfigManMIB
.ciscoConfigMan
MIBObjects.ccmHistory.ccmHistoryEventTable.ccmHistoryEventEntry.ccmHistoryE
ventConfigDesti nation.6090--&gt;3
</Description>
  <DetectionTime type="java.util.Date">Mon Aug 02 11:47:00 IDT
  2010</DetectionTime>
  <DetectionTypeEnum type="Integer">3</DetectionTypeEnum>
  <DuplicationCount type="Integer">1</DuplicationCount>
  <Enterprise type="String">.1.3.6.1.4.1.9.9.43.2</Enterprise>
  <ErrorStatusEnum type="Integer">0</ErrorStatusEnum>
  <Name type="Integer">1230</Name>
  <ReductionCount type="Integer">1</ReductionCount>
  <SeverityEnum type="Integer">1</SeverityEnum>
  <Source
  type="Oid">{[ManagedElement(Key=169.254.192.231)][Trap]}</Source>
  <State type="String">Cisco Configuration management event
  notification</State>
  <TranslatedEnterprise
  type="String">.iso.org.dod.internet.private.enterprises.cisco.ciscoMgmt.
  ciscoConfigManMIB.
  ciscoConfigManMIBNotificationPrefix</TranslatedEnterprise>
  <TrapTypeOid type="String">.1.3.6.1.4.1.9.9.43.2.0.1</TrapTypeOid>
  <TrapValues type="IMObjects_Array">
        <ITrapValue type="ITrapValue" instance_id="2">
              <ID type="Oid">{[TrapValue(Location=0)]}</ID>
              <SnmpOid type="String">.1.3.6.1.2.1.1.3.0</SnmpOid>
              <SnmpTranslatedOid
```

```
            type="String">.iso.org.dod.internet.mgmt.mib-
            2.system.sysUpTime.sysUpTimeInstance</SnmpTra nslatedOid>
            <SnmpTranslatedValue type="String">159 days, 0 hours, 56
            minutes, 28 seconds.
            </SnmpTranslatedValue>
            <SnmpValue type="String">159 days, 0 hours, 56 minutes, 28
            seconds. </SnmpValue>
        </ITrapValue>
      <ITrapValue type="ITrapValue" instance_id="3">
        <ID type="Oid">{[TrapValue(Location=1)]}</ID>
        <SnmpOid type="String">.1.3.6.1.6.3.1.1.4.1.0</SnmpOid>
        <SnmpTranslatedOid
        type="String">.iso.org.dod.internet.snmpV2.snmpModules.snmpMIB.snm
        pMIBObjects.snmpTrap.snm pTrapOID.0</SnmpTranslatedOid>
        <SnmpTranslatedValue
        type="String">.1.3.6.1.4.1.9.9.43.2.0.1</SnmpTranslatedValue>
        <SnmpValue type="String">.1.3.6.1.4.1.9.9.43.2.0.1</SnmpValue>
      </ITrapValue>
    <ITrapValue type="ITrapValue" instance_id="4">
        <ID type="Oid">{[TrapValue(Location=2)]}</ID>
        <SnmpOid
        type="String">.1.3.6.1.4.1.9.9.43.1.1.6.1.3.6090</SnmpOid>
        <SnmpTranslatedOid
        type="String">.iso.org.dod.internet.private.enterprises.cisco.cisc
        oMgmt.ciscoConfigManMIB.
        ciscoConfigManMIBObjects.ccmHistory.ccmHistoryEventTable.ccmHistor
        yEventEntry.ccmHistoryEv entCommandSource.6090</SnmpTranslatedOid>
        <SnmpTranslatedValue type="String">1</SnmpTranslatedValue>
        <SnmpValue type="String">1</SnmpValue>
  </ITrapValue>
  <ITrapValue type="ITrapValue" instance_id="5">
        <ID type="Oid">{[TrapValue(Location=3)]}</ID>
        <SnmpOid
        type="String">.1.3.6.1.4.1.9.9.43.1.1.6.1.4.6090</SnmpOid>
        <SnmpTranslatedOid
        type="String">.iso.org.dod.internet.private.enterprises.cisco.cisc
        oMgmt.ciscoConfigManMIB.
        ciscoConfigManMIBObjects.ccmHistory.ccmHistoryEventTable.ccmHistor
        yEventEntry.ccmHistoryEv entConfigSource.6090</SnmpTranslatedOid>
        <SnmpTranslatedValue type="String">2</SnmpTranslatedValue>
        <SnmpValue type="String">2</SnmpValue>
    </ITrapValue>
    <ITrapValue type="ITrapValue" instance_id="6">
        <ID type="Oid">{[TrapValue(Location=4)]}</ID>
        <SnmpOid
        type="String">.1.3.6.1.4.1.9.9.43.1.1.6.1.5.6090</SnmpOid>
        <SnmpTranslatedOid
        type="String">.iso.org.dod.internet.private.enterprises.cisco.cisc
        oMgmt.ciscoConfigManMIB.
        ciscoConfigManMIBObjects.ccmHistory.ccmHistoryEventTable.ccmHistor
        yEventEntry.ccmHistoryEv
        entConfigDestination.6090</SnmpTranslatedOid>
        <SnmpTranslatedValue type="String">3</SnmpTranslatedValue>
        <SnmpValue type="String">3</SnmpValue>
    </ITrapValue>
  </TrapValues>
  </Imo>
```

```
</IObjectCreateNotification>
</IMObjects_Array>
```

3.4 BQL Commands

BQL supports the following basic commands, which are supported by most IMOs:

- **Create**—Create a new IMO. See Create Command, page 48.
- **Get**—Retrieve information for a specific object. See Get Command, page 50.
- **Register**—Register for change notifications. See Register Command, page 52.
- **Update**—Set properties and relations for an existing object. See Update Command, page 54.
- **Refresh**—Retrieve data, on demand, from the NE before polling its model. See Refresh Command, page 56.
- **Delete**—Delete an existing IMO. See Delete Command, page 57.

The following sections describe these basic commands. There are other specialized commands, such as **Activation** commands, that perform operations that cannot be mapped into one of these basic commands. These specialized commands are beyond the scope of this document.

The following terms are used to define the BQL commands:

- VNE IMOs—All inventory (physical and logical) IMOs that are within the context of a device or VNE.
- Business IMOs—IMOs that are related to Virtual Private Network (VPN), Virtual LANs (VLAN), Ethernet virtual Connection (EVC), and so on.
- AVM and VNE management IMOs—IMOs that are related to AVM and VNE management.
- Fault IMOs—IMOs that are related to events, alarms, and tickets.

3.4.1 BQL Command Format

After the BQL server session is authenticated, all commands and responses are formatted as plain-text XML messages. Each command has the following format:

```
<command name="commandName">
  <param name="paramName1">
        <value>ParamValue1</value>
  </param>
  <param name="paramName2">
        <value>ParamValue2</value>
  </param>
  ...
</command>
.
```

Every command must end with an EOT sequence, which is a *<NewLine>* character (“\n”) followed by a dot in an empty line, as follows:

```
EOT:\n.\n
```

Therefore, every command has the format:

```
<command name="commandName">
...
</command>\n.\n
```

The EOT sequence also terminates all BQL replies. However, when receiving unsolicited notifications, the EOT sequence changes to the format \n$\n. For more details, see Processing BQL Notification Messages, page 87.
Figure 3-7 contains a sample BQL command example.

Figure 3-7 BQL Get Command Example

- Command Name—BQL commands, such as **Get**, **Update**, **Create**, and so on.
- OID—Unique identifier of IMO in the system. For more details, see IMO OIDs, page 8.
- Retrieval Specification—Defines the IMO to be retrieved. For more details, see Retrieval Specification, page 14.

The following is the list of XML escape characters that are supported:

Code	Displays as
&amp	&
&lt	<
&gt	>
&quot	“
&apos	‘

3.4.2 Create Command

Description

The **Create** command creates a new data object in the system. The command the data of the new IMO as a parameter.

You can run this command on the business, AVM, and VNE management IMOs. You cannot run this command on the VNE and fault IMOs. See the *Cisco Prime Network Information Model* Javadoc on the Prime Network Technology Center website for further details.

Syntax

Table 3-6 BQL Create Command Syntax

Element	Details
CREATE_COMMAND	<command name=”Create”> <param name=”imobject”> <value> IMO_DATA </value> </param> </command>
IMO Data	IMO Representations

See BQL Command Format, page 47 to understand the BQL command format and syntax.

Output

A dot (.) indicates success. In the event of a failure, an error message is displayed.

Examples

The following example shows the usage of the BQL **Create** command to add a unit in the Prime Network gateway. The unit IP address (10.77.213.238) is added under the default protection group with the high availability feature enabled.

```
<command name="Create">
  <param name="imobject">
        <value>
        <management.IMC>
              <ID type="Oid">
                    {[MCNetwork][MCVM(IP=10.77.213.238)]}
              </ID>
            <HighAvailabilityEnabled
            type="Boolean">true</HighAvailabilityEnabled>
            <ProtectionGroupOid type="Oid">
           {[MCNetwork][MCVM(IP=10.77.213.238)][ProtectionGroup(Key=defaul
           t-pg)]}
           </ProtectionGroupOid>
        </management.IMC>
      </value>
  </param>
</command>
.
```

3.4.3 Get Command

Description

The **Get** command retrieves an IMO construct (related IMOs) based on a specified RS. It also enables registering for notifications of changes in the specified IMOs.

You can run this command on all IMOs. See the *Cisco Prime Network Information Model* Javadoc on the Prime Network Technology Center website for further details.

In case you run a Get on a VNE, you can choose not to retrieve the IMO, and only receive the notifications by setting the emptyInitialImo parameter in the retrieval spec to true. You can also choose to receive notifications with change or without change by setting the emptyNotifications parameter in the retrieval spec to true.

Syntax

Table 3-7 BQL Get Command Syntax

Element	Details
GET_COMMAND	<command name=”Get”> <param name=”oid”> <value> OID </value> </param> <param name=”rs”> <value> RETRIEVAL_SPEC </value> </param> </command>
OID	OID format is explained in IMO OIDs, page 8
RETREIVAL_SPEC	RS format is explained in Format and Syntax of Aspect and Retrieval Specification, on page 14.

See BQL Command Format, page 47 to understand the BQL command format and syntax.

It is possible to execute a Get command multiple times on the same OID and with the same retrieval spec. To get notified about the already existing and active registration for the same object instance with the same retrieval spec, execute the following command:

```
unregisterSameActiveCommand true
```

This command prevents you from running duplicate Get commands on the same object instance with the same retrieval spec.

Change the parameter value to false (as given below), to run duplicate Get commands for the same OID with the same retrieval spec.

```
unregisterSameActiveCommand false
```

Output

The output consists of IMO data in XML format. If the **Get** command registers for changes, the output terminates with \n$\n. Otherwise, the output of the command ends with the standard EOT sequence \n.\n.

Examples

The following example shows the usage of the BQL **Get** command to retrieve the network element (core-crs1-p2) properties.

```
<command name="Get">
  <param name="oid">
        <value>{[ManagedElement(Key=core-crs1-p2)]}</value>
  </param>
  <param name="rs">
    <value>
        <key name="NE report">
        <entry name="register">false</entry>
        <key name="requiredProperties">
        <key name="com.sheer.imo.IManagedElement">
        <entry name="*"/>
        </key>
        </key>
        </key>
      </value>
  </param>
</command>
.
```

Keep-Alive Notification

Get command registered for changes (register=”true”) can send a Keep-Alive notification in specified time interval in order to ensure that the request and registration are alive and valid. The client should request this notification by adding “keepAlive” entry to Retrieval Specification parameter. “keepAlive” may have a true/false value.

The keep-alive notification interval can be specified in Registry and it is the same for all Get commands. The entry name is "mmvm/commands/specialCommands/com.sheer.framework.commands.Get/keep-alive-interval". Default value is 30000.

Example for CID of Get command with keepAlive

```
<command name="Get">
        <param name="oid">
                <value>{[ManagedElementSnapshot]}</value>
        </param>
<param name="rs">
<value>
<key name="Get And KeepAlive">
    <entry name="depth">100</entry>
    <entry name="register">true</entry>
                <entry name="keepAlive">true</entry>
    <entry name="cachedResultAcceptable">true</entry>
    <key name="requiredProperties">
        <key name="*">
            <entry name="*"/>
        </key>
    </key>
</key>
</value>
</param>
</command>
.
```

Keep-Alive Notification Example:

```
<IMObjects_Array>
  <IKeepAliveNotification type="IKeepAliveNotification" instance_id="0">
    <ID type="Oid">{[Notification(Time=1327241157540)]}</ID>
  </IKeepAliveNotification>
</IMObjects_Array>
```

3.4.4 Register Command

Description

The **Register** command registers for change notifications on any data scope.

Note The Register command is not supported on IMOs implemented by the VNE.

Syntax

Table 3-8 BQL Register Command Syntax

Element	Details
REGISTER_COMMAND	<command name=”Register”> <param name=”oid”> <value> OID </value> </param> <param name=”rs”>

Element	Details
	<value> RETRIEVAL_SPEC </value> </param> </command>
OID	OID format is explained in IMO OIDs, page 8
RETREIVAL_SPEC	RS format is explained in Format and Syntax of Aspect and Retrieval Specification, on page 14.

To understand the BQL command format and syntax, see BQL Command Format, page 47.

It is possible to execute a Register command multiple times on the same OID and with the same retrieval spec. To get notified about the already existing and active registration for the same object instance with the same retrieval spec, execute the following command:

```
unregisterSameActiveCommand true
```

This command prevents you from running duplicate Register commands on the same object instance with the same retrieval spec.
Change the parameter value to false (as given below), to register multiple notifications for the same OID with the same retrieval spec.

```
unregisterSameActiveCommand false
```

Output
A dot (.) indicates success. In the event of a failure, an error message is displayed.

Examples

In the following example, the client application is registering for notification of ticket updates for all network elements that are managed in Prime Network. Here, the **Register** command is used. When you run this command, the client application is registered to receive notification without retrieving the data.

```
<command name="Register">
<param name="oid">
<value>{[TicketListAspect]}</value>
</param>
<param name="rs">
<value>
<key name="">
  <entry name="depth">100</entry>
  <entry name="register">true</entry>
```

```
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.newalarm.ITicketListAspect">
              <entry name="Tickets" />
        </key>
      <key name="com.sheer.imo.newalarm.ITicket">
        <entry name="*"/>
      </key>
  </key>
</key>
</value>
</param>
</command>
.
```

3.4.5 Update Command

Description

The **Update** command modifies an existing IMO.

This command receives an OID to modify an array of notifications of type scalar, add notification, and remove notification. These notifications describe how to modify the IMO; that is, which properties to change, add, or remove.

You can run this command on the business, AVM, and VNE management IMOs. You cannot run this command on the VNE and fault IMOs. See the *Cisco Prime Network Information Model* Javadoc on the Prime Network Technology Center website for further details.

Syntax

Table 3-9 BQL Update Command Syntax

Element	Details
UPDATE_COMMAND:	<command name=”Update”> <param name=”oid”> <value> OID </value> </param> <param name=”imoobjectArr”> <value> NOTIFICATION_ARRAY </value> </param> </command>
OID:	OID format is explained in IMO OIDs, page 8

Element	Details
NOTIFICATION_ARRAY:	NOTIFICATION_IMO | NOTIFICATION_IMO NOTIFICATION_ARRAY
NOTIFICATION_OBJECT:	IMO of type IScalarNotification, IAddNotification, or IRemoveNotification as described in Understanding Notification Messages, page 92

See BQL Command Format, page 47 to understand the BQL command format and syntax.

Output

A dot (.) indicates success. In the event of a failure, an error message is displayed.

Examples

The following example shows the usage of the BQL **Update** command to edit a business tag attached to a network element (PE-209_Sim).

```
<command name="Update">
  <param name="oid">
        <value>{[ManagedElement(Key=PE-
        209_Sim)][PhysicalRoot][Chassis][Slot(SlotNum=0)][Modul
        e][Port(PortNumber=FastEthernet0/0)][BusinessObject]}</value>
  </param>
  <param name="imobjectArr">
        <value>
              <IScalarNotification>
              <ID type="Oid">{[Notification]}</ID>
              <NewIMO type="IBusinessObject">
              <ID
              type="Oid">{[ManagedElement(Key=PE-
              209_Sim)][PhysicalRoot][Chassis][Slot(SlotNum=0
              )][Module][Port(PortNumber=FastEthernet0/0)][BusinessObject
              ]}</ID>
              <Name type="String">Hello World again</Name>
              </NewIMO>
              <PropertyName type="String">Name</PropertyName>
              </IScalarNotification>
        </value>
  </param>
</command>
.
```

3.4.6 Refresh Command

Description

The IMO polls its data from the VNE model; the VNE model polls its data from the NEs. When a **Get** command is executed, a set of IMOs is retrieved. Sometimes the data is not updated to the current NE state when it was polled from the VNE model. When the most recent data is needed, use the **Refresh** command to request the VNE to "retrieve now" the data from the NE before polling its model.

You can use the **Refresh** command for VNE IMOs. This command sends a message to the relevant DC in the VNE model to refresh all its registrations. Once the NE replies with the information, the IMO is refreshed. A notification message is sent if you have registered. See the *Cisco Prime Network Information Model* Javadoc on the Prime Network Technology Center website for further details.

Syntax

Table 3-10 BQL Refresh Command Syntax

Element	Details
REFRESH_COMMAND	<command name="Refresh"> <param name="oid"> <value> OID </value> </param> </command>
OID:	OID format is explained in IMO OIDs, page 8

See BQL Command Format, page 47 to understand the BQL command format and syntax.

Output

A dot (.) indicates success. In the event of a failure, an error message is displayed.

Examples

The following example shows the usage of the BQL **Refresh** command to refresh the physical inventory data of a network element.

```
<command name="Refresh">
  <param name="oid">
        <value>{[ManagedElement(Key=PE-209_Sim)]}</value>
  </param>
  <param name="oids">
    <value>{[ManagedElement(Key=PE-
    209_Sim)][PhysicalRoot][Chassis][Slot(SlotNum=1)][Module][P
```

```
    ort(PortNumber=FastEthernet1/0)][PhysicalLayer]}</value>
  </param>
</command>
```

3.4.7 Delete Command

Description

The **Delete** command deletes an existing IMO. The deleted IMO is identified by its OID. Not all data elements in the system can be deleted.

You can run this command on the business, AVM, and VNE management IMOs. You cannot run this command on the VNE and fault IMOs. See the *Cisco Prime Network Information Model* Javadoc on the Prime Network Technology Center website for further details.

Syntax

Table 3-11 BQL Delete Command Syntax

Element	Details
DELETE_COMMAND	<command name=”Delete”> <param name=”oid”> <value> OID </value> </param> </command>
OID:	OID format is explained in IMO OIDs, page 8

See BQL Command Format, page 47 to understand the BQL command format and syntax.

Output

A dot (.) indicates success. In the event of a failure, an error message is displayed.

Examples

The following example shows the usage of the BQL **Delete** command to delete a user-defined report.

```
<?xml version="1.0" encoding="UTF-8"?>
  <command name="Delete">
        <param name="oids">
            <value>{[ReportList(ListTargetOid={[ReportRoot][ReportCategory
            (Category=Events Reports)][ReportType(Type=Prime Network vs.
            Event Archive Statistics)]})][Report(Id=9)]}</value>
        </param>
  </command>
```

3.4.8 BQL Output

Since Prime Network 3.8, the BQL results include a new “instance_id” attribute for each IMO, as shown in bold in the following example. (The BQL command syntax is unchanged.)

```
<CommandResult>
  <IManagedElement type="IManagedElement" instance_id="0">
        <ID type="Oid">{[ManagedElement(Key=C7-SW8)]}</ID>
        <PhysicalRoot type="IPhysicalRoot" instance_id="3">
            <ID type="Oid">{[ManagedElement(Key=C7-
            SW8)][PhysicalRoot]}</ID>
            <EquipmentHolders type="IMObjects_Array">
              <IChassis type="IChassis" instance_id="4">
              <ID type="Oid">{[ManagedElement(Key=C7-
              SW8)][PhysicalRoot][Chassis]}</ID>
              </IChassis>
            </EquipmentHolders>
       </PhysicalRoot>
       <DeviceName type="String">C7-SW8</DeviceName>
       <LogicalRoot type="ILogicalRoot" instance_id="5">
         <ID type="Oid">{[ManagedElement(Key=C7-SW8)][LogicalRoot]}</ID>
        </LogicalRoot>
        <ElementType type="String">Cisco Catalyst 3750ME</ElementType>
        <IP type="com.sheer.types.IPAddress">10.56.101.37</IP>
        <VendorEnum type="Integer">3</VendorEnum>
        <ElementTypeKey
        type="String">CISCO_CATALYST_3750ME</ElementTypeKey>
  </IManagedElement>
</CommandResult>
```

The numeration values that are not available in the device will be returned as MIN_INTEGER (-2147483648). This value will not be displayed in the GUI, and all OSS applications should treat this value as Not Available or Unknown. The example provided below shows the output of the port properties which includes enums like AutoNegotiateEnum, InputFlowControlEnum etc.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Get">
    <param name="oid">

<value>{[ManagedElement(Key=49_6151_13_7YJ3_FMED32_AGS1)][PhysicalRoot][Ch
assis][Slot(SlotNum=0)][Module][Port(PortNumber=TenGigE0/0/0/0)]}</value>
    </param>
    <param name="rs">
        <value><key name="PortConnectorController">
    <entry name="depth">0</entry>
    <entry name="register">true</entry>
    <entry name="cachedResultAcceptable">false</entry>
    <key name="requiredProperties">
        <key name="*">
            <entry name="*"/>
```

```
        </key>
    </key>
    <key name="excludedProperties">
        .
        .
        .
        .
        .


<AggregationGroup
type="Oid">{[ManagedElement(Key=49_6151_13_7YJ3_FMED32_AGS1)][LogicalRoot]
[Context(ContextName=Default
context)][DataLinkAggregationContainer(Type=1)][DataLinkAggregation(Index=
24)]}</AggregationGroup>

====================================
          <AutoNegotiateEnum type="Integer">-
2147483648</AutoNegotiateEnum>
====================================
          <ContainedCurrentCTPs type="IMObjects_Array">
            <technologies.IVlanEncapMux type="technologies.IVlanEncapMux"
instance_id="3">
              <ID

              .
              .
              .
              .
              <TypeEnum type="Integer">6</TypeEnum>
            </technologies.IVlanEncapMux>
          </ContainedCurrentCTPs>
          <DiscoveryProtocols type="">Null</DiscoveryProtocols>

====================================
          <DuplexEnum type="Integer">-2147483648</DuplexEnum>
====================================
          <Efps type="">Null</Efps>

====================================
          <InputFlowControlEnum type="Integer">-
2147483648</InputFlowControlEnum>
====================================

          <IsELMIEnabled type="Boolean">false</IsELMIEnabled>
          <MacAddress type="com.sheer.types.MacAddress">8C B6 4F E8 2F
92</MacAddress>
          <OAMAdminStatus type="">Null</OAMAdminStatus>

====================================
          <OutputFlowControlEnum type="Integer">-
2147483648</OutputFlowControlEnum>
====================================

          <ScriptMetadataOids type="">Null</ScriptMetadataOids>
          <TypeEnum type="Integer">380</TypeEnum>
        </technologies.IEthernet>
```

```
</ContainedCurrentCTPs>
<ContainingTPs type="IMObjects_Array">
  <IPortConnector type="IPortConnector" instance_id="12">
                  <ID
```

4 BQL Implementation

This section contains the following topics:

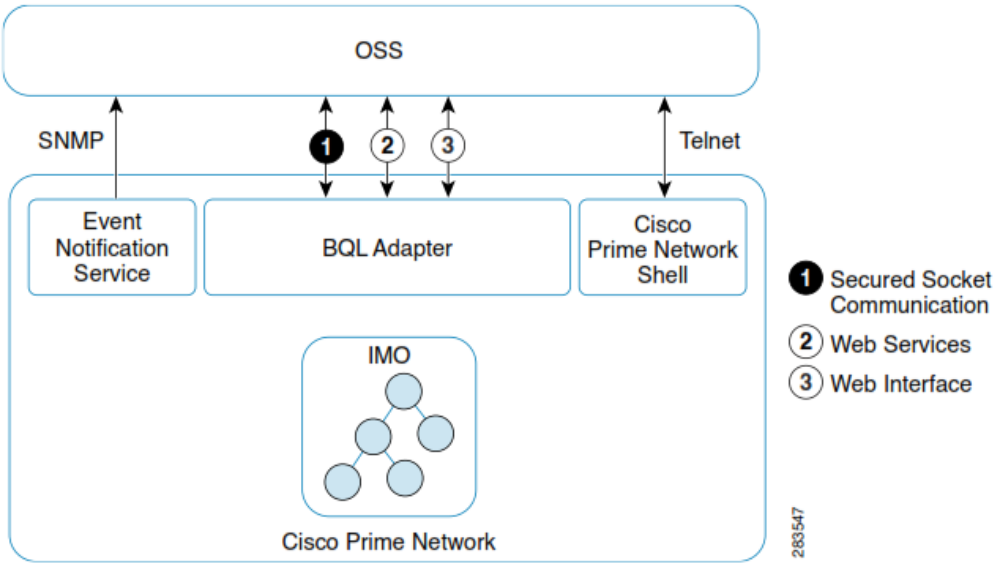
- Running BQL Using Secured Socket Communication
- Running BQL Using Web Services
- Running BQL using the Web Interface

4.1 Running BQL Using Secured Socket Communication

The Cisco Prime Network gateway and remote clients communicate over TCP sockets. All data exchanged between the Prime Network gateway and clients is formatted as XML messages containing data objects (see Understanding IMO, page 6).

Figure 4-1 illustrates how an OSS application communicates with the BQL adapter over secured socket.

Figure 4-1 Prime Network Secured Socket Communication



4.1.1 Connecting to the BQL Adapter

The BQL interface uses either TCP or SSL to access an OSS application.

Connecting to the BQL Adapter over TCP

The BQL interface uses a TCP socket transport for local access. The BQL server on the Prime Network gateway listens to the incoming connection attempts on a well-known port, as follows:

- Protocol—TCP
- Port number—9002
- Socket type—Streamable
- Bind—localhost

You can use TCP socket transport for remote access. However, by default, the remote access is disabled through TCP on port 9002. To enable TCP socket for remote access, see BQL Server, page 66.

Connecting to the BQL Adapter over SSL

- The BQL interface uses Secure Socket Layer (SSL) transport for remote access. The BQL adapter on the Prime Network gateway listens to the incoming connection attempts on a well-known port, as follows: Protocol—SSL
- Port number—9003
- Socket type—Streamable
- Bind—remote

By default, the remote access is enabled through SSL on port 9003 for the BQL interface.

4.1.2 Opening and Closing a BQL Adapter Session

The connecting application must first log into the Prime Network gateway using SSH and then use Telnet on port 9002 to open the TCP socket. For remote access applications, you must use a secured socket on port 9003. For more details, see Connecting to the BQL Adapter, page 60.

The connecting application must authenticate the session using a username and password (which have been previously defined within Prime Network). The authentication command is:

```
openconnection user=<Username> password=<Password>\n.\n
```

Note Using the Prime Network GUI, you must create one or more users whose roles allow them to access the Prime Network integration interfaces. When a BQL server session is opened, the scopes associated with the user profile are applied to the session. See the *Cisco Prime Network 4.2.2 Administrator Guide* for more information about user scopes.

To disconnect from the BQL interface, enter the command:

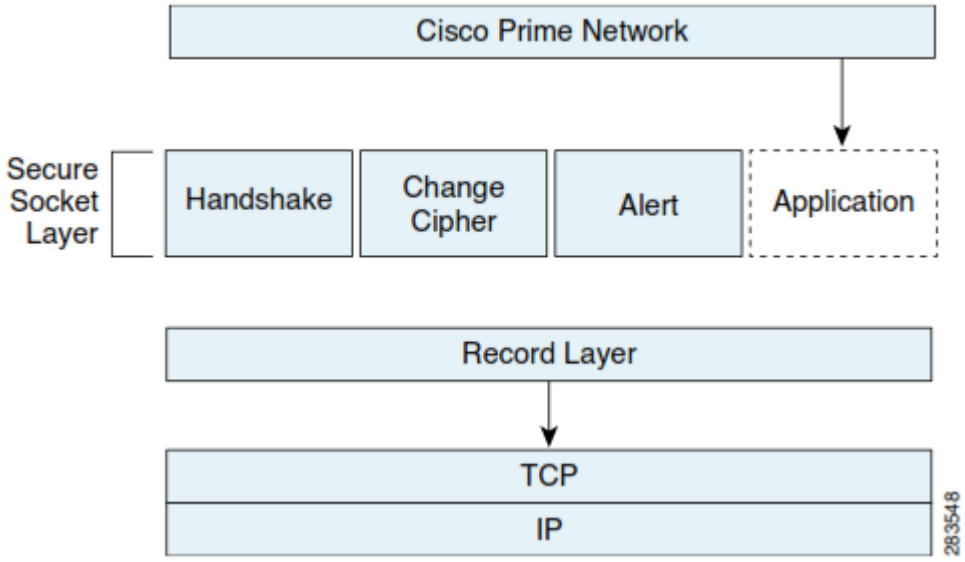
```
exit\n.\n
```

Then close the connection to the Prime Network gateway.

4.1.3 Understanding the BQL Secured Socket Communication Architecture

The Secure Socket Layer (SSL) sits between the TCP layer and the application in the Internet protocol architecture (see Figure 4-2).

Figure 4-2 SSL Protocol Architecture



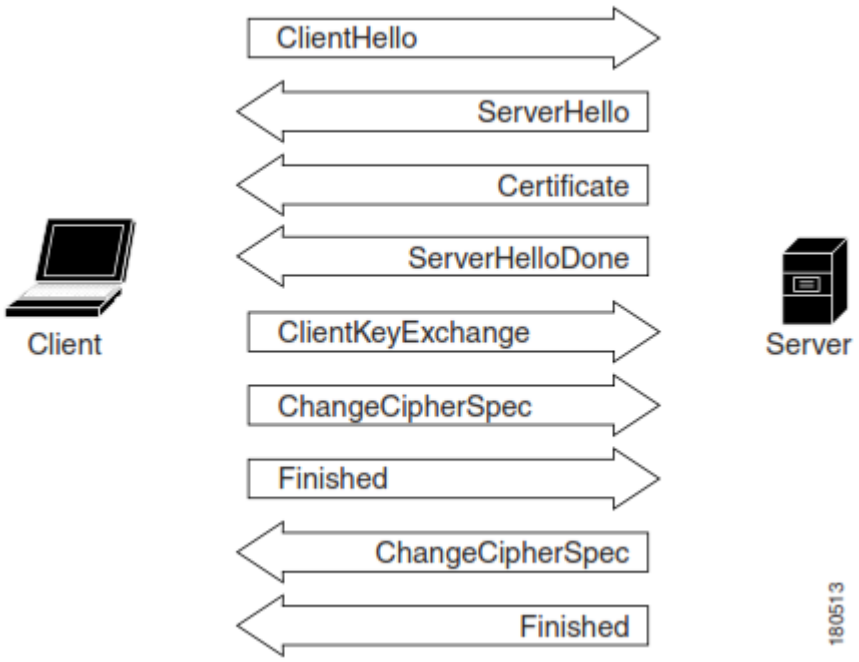
The SSL protocol architecture includes the following components:

- Record layer—Provides framing and padding for the other SSL components, and for the application. In addition, it identifies the SSL component being used in the message. It is common to all SSL messages.
- Handshake—Responsible for the negotiations that start an SSL session.
- Change Cipher—Activates the negotiated security parameters for a session.
- Alert—Enables parties to exchange error or warning information. In addition, it identifies problems with the protocol or potential security problems with the session.
- Application—Communicates with the record layer which coordinates the TCP socket interface.

Negotiation Process

All SSL sessions are established between a client and a server.

Figure 4-3 SSL Negotiation



The client starts the exchange by sending a *Client Hello* message to the server to let the server know that the client wants to establish secure communications. It also proposes a list of security parameters (cipher suites) that the client would like to use.

A *Server Hello* message is sent by the server, informing the client of the preferred cipher suite. In addition, it tells the client that the server is willing to proceed with the SSL negotiation. The server follows this immediately with a *Certificate* message which carries the server's public key certificate.

The server then sends a *Server Hello Done* message to the client, telling the client that the server has finished its part of the initial negotiations.

The client generates random numbers to use as a shared session key. In a *Client Key Exchange* message, the client encrypts the session key with the server's public key and sends the result to the server. The client depends on the server's ability to decipher the Client Key Exchange message to verify the server's identity.

The client then sends a *Change Cipher Spec* message that tells the server to activate the negotiated cipher suite. From that point on, all messages from the client are encrypted using the algorithm from the Server Hello and the session key in the Client Key Exchange. The client sends an encrypted *Finished* message, which ensures that both parties use the same cryptographic algorithms and parameters. The negotiation is considered to be successful when the message that follows the client's Change Cipher Spec is successfully decrypted as a Finished message.

The server sends a Change Cipher Spec message that tells the client that all subsequent messages from the server will use the negotiated security. The server then sends its own Finished message, which enables the client to confirm that the negotiated security is in place.

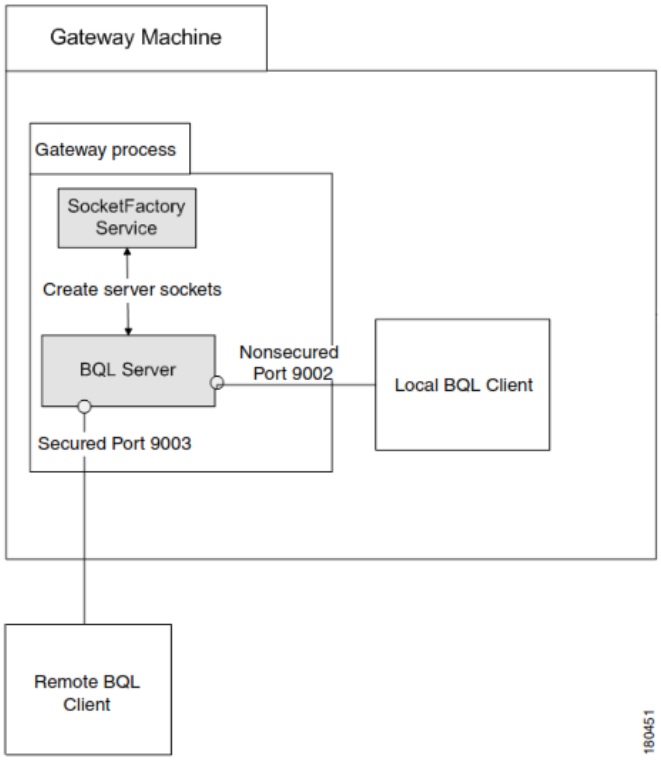
BQL Adapter and SSL Socket

The secured BQL interface uses a socket factory service that implements encrypted SSL sockets. The sockets are used to facilitate secure communication between the gateway and BQL clients. The socket factory service creates a regular server socket on port 9002 for the BQL server for local access and a secured server socket on port 9003 for remote access. Port 9002 can also be configured to allow or not allow unsecured access for remote clients for debugging purposes (the default is to not allow unsecured access). The gateway allows unsecured and secured connections from the local host and only secured connections from other machines.

With SSL version 3.0, SSL keys are created at installation time. SSL keys can also be recreated (see

Perl Example to Generate the SSL Key, page 69). All secured connections use the same keys. The gateway holds both the public and private keys while the BQL clients have only the public key. The keys are used only for encryption and not for authentication.

Figure 4-4 System Components



In Figure 4-4, the BQL server allows BQL communication between the server and clients, using encrypted communication (SSL) for remote connections and standard communication for local connections. The inputs are client connection requests, and the outputs are secured or nonsecured connections. For more information, see BQL Server, page 66.

The socket factory service creates the SSL-secured and nonsecured sockets. The inputs are BQL server requests for SSL-secured sockets and nonsecured sockets; the outputs are SSL-secured sockets and nonsecured sockets. For more information, see Socket Factory **Service**, page 67.
There are two ways to access the BQL server:

- Negotiate with the server and get the public key during the negotiation.
- If the public key is not provided explicitly, negotiate with the server using the process described in

- Figure 4-3.

When the private or public key is created on a gateway machine, it is created in a certificate file that uses the .cer format. However, when it is accessed from the client machine that uses the Perl package, it needs to use the .pem format. The .pem format is generated from the .cer format.

The sheer.pem or client-cert.pem file that is used by the client machine is created by running a utility called keytool on the server (an automatic script for doing this is located on the server itself). To create it, run the following command:

```
keytool -export -rfc -alias ana -keystore .keystore -file client-cert.pem -
storepass
$PASSWORD
```

After you export the server certificates into the .cer file, download it to the client machine. Use the command keytool to generate the private key .keystore and .trustore files from the .cer file:

```
keytool -genkey -alias ana -keyalg RSA -dname "$DNAME" -keypass $PASSWORD
-keystore
.keystore -storepass $PASSWORD

keytool -export -alias ana -keystore .keystore -file sheer.cer -storepass
$PASSWORD

keytool -import -alias ana -keystore .trustore -file sheer.cer -keypass
$PASSWORD
-storepass $PASSWORD
```

The client-cert.pem file is then created as follows:

```
Sheer.cer
|         |
.keystore .trustore
|
client-cert.pem
```

The names of the files are provided in the command line itself (hard-coded).

The .keystore and .trustore files are located in
NETWORKHOME/Main/resourcebundle/com/sheer;
where, *NETWORKHOME* is the Prime Network installation directory.

BQL Server

The SSL certificate that is used by all sockets is managed by the gateway and stored in a local file. Clients that want to validate the server's public key should have this file locally and supply it in the handshake process.
The registry entry allowRemoteUnsecured allows remote access that is not secured for debugging purposes.
Possible values:

- true—The BQL server allows remote clients to connect to the unsecured port (9002).
- false—The BQL server allows only local clients to connect to the unsecured port (9002). The default value is false.

To set the value to true:

1. Log into the Prime Network gateway as *network-user* (where *network-user* is the operating system user for the Prime Network application, created when Prime Network is installed; an example of *network-user* is network310).
   Change to the Main directory:
   `# cd $ANAHOME/Main/` (Where $*ANAHOME* is the Prime Network installation directory.)
2. Run the following commands:

   Note Changes to the registry should only be carried out with the support of Cisco. For details, contact your Cisco account representative.

```
runRegTool.sh -gs 127.0.0.1 add 0.0.0.0
avm11/services/bqlServer/
runRegTool.sh -gs 127.0.0.1 set 0.0.0.0
avm11/services/bqlServer/allowRemoteUnsecured true
```

3. Restart the Prime Network gateway using the command
   `anactl restart`

The script createSSLKeys (under *NETWORKHOME*/Main/scripts) is used when a new gateway is installed. This script creates the keys necessary for encryption.
When there is a standard SSL connection, the client starts to negotiate with the server. The client does not require any key on the client side; the client gets the public key from the server during the initial negotiation.

The following entry is used to open the key files stored in the file security.properties located in
NETWORKHOME/Main/resourcebundle/com/sheer/:

```
SSLPassword
```

Socket Factory Service

The following registry entries are used to create SSL-secured sockets:

- keystoreType—Format of the file holding the public and private keys (certificates).
- securityAlgorithm—Sun machine (SunX509) or IBM machine (IBMX509).
- securityProtocol—The security protocol; that is, Transport Layer Security (TLS).

Example

```
<key name="socketfactory">
  <entry name="keystoreType">jceks</entry>
  <entry name="securityAlgorithm">SunX509</entry>
  <entry name="securityProtocol">TLS</entry>
</key>
```

4.1.3.1 Perl Example to Generate the SSL Key

In this example, the client receives the server public key in one of the following ways:

- During negotiation
- At the command line

Note When executing a script, to specify the certificate (public key) in the script (where no negotiation is required), the --certificate entry uses a hard-coded filename and relative path. It uses the Keys\\client-cert.pem file to take the relevant SSL connection parameters. This file should be transferred manually from the server (where it was created) to the specific client machine.

This script executes a Sheer-1 script on the gateway. It receives the device key as it appears in Prime Network, and two files. The two files contain:

- The BQL statement number. The parameters are in the format *$param name$* and the device key is replaced with ---interface---.
- Parameters in the order in which they appear in the BQL statement.

Note
- The gateway IP address is hard-coded within the script; see the gateway_ip parameter.
- The gateway username and password are hard-coded within the script; see the login parameter.

```
my ($gateway_ip, $user, $pass, $device_key, $params_input, $xml_input,
$port, $ssl_mode,
$certificate)=("localhost","", "", "","", "", 0, 0, 0);
my $bql_statement;

GetOptions(
   'ip=s'       => \$gateway_ip,
   'user=s'     => \$user,
   'password=s' => \$pass,
   'key=s'      => \$device_key,
   'params=s'   => \$params_input,
   'file=s'     => \$xml_input,
   'ssl'        => \$ssl_mode,
   'port=s'     => \$port,
   'certificate'=> \$certificate,
);

usage() if not $gateway_ip or not $user or not $pass;


#print_params();
read_params_file();
read_input_file();
open_socket($ssl_mode);
exit(0);
```

Functions

```
sub usage
{
  print STDERR << "EOF";

  usage: $0 [-ip Gateway ip Address] [-user user] [-password password] [-
  key device key] [-params params input file]
        [-f xml input file] --ssl --certificate

  -ip          : Host Gateway ip address
  -user        : User name
  -password    : Password
  -key         : device key
  -params      : params input file name
  -f           : XML Input file name
  -ssl         : ssl mode
  -certificate : in case we want to work in mode where the certificate is
taken from a local pem format file located on the pc itself.

  example: $0 -ip 10.56.22.196 -user gw_user -password gw_password -key
CRS-PE7 -params
SP-VRF-INSTANCE.ini
        -f SP-VRF-INSTANCE.txt --ssl -certificate
EOF
  exit;

# read parameters file
my @params_list;

sub read_params_file {
#   open (my $handle_params_input, "<", "$params_input") or die "cannot
open parameters file $params_input\n";
  open (my $handle_params_input, "<", "$params_input") or return;
  while (my $row = <$handle_params_input>) {
        #print "row=$row";
        push @params_list, $row;
  };
  close $handle_params_input;
}


# read xml input file
sub read_input_file {
  open (my $handle_xml_input, "<", "$xml_input") or die "cannot open xml
  input file
$xml_input\n";

  while (<$handle_xml_input>) {
  my $row = $_;
  if (/\<value\>\{\[ManagedElement\(Key=\-\-\-interface\-\-\-
  \)\]\}\<\/value\>/) {
        $row =~
s/\<value\>\{\[ManagedElement\(Key=\-\-\-interface\-\-\-
\)\]\}\<\/value\>/\<value\>\{\[Man
agedElement\(Key=$device_key\)\]\}\<\/value\>/g;
}
  elsif (/\<value\>\$([a-zA-Z0-9_\-]*)\$\<\/value\>/) {
```

```
        my $param = shift @params_list;
        chomp $param;
        #   print "Param: $param\n";
        $row =~ s/\<value\>\$([a-zA-Z0-9_\-
  ]*)\$\<\/value\>/\<value\>$param\<\/value\>/g;
}
  $bql_statement .= $row;
}
  $bql_statement .= "\n.\n";

  print "BQL
Statement:\n
  close $handle_xml_input;
}
```

Open Socket and Login for Prompt

```
sub open_socket {
  my ($ssl) = @_;

  my $socket;
  if ($ssl) {
        eval {
              require IO::Socket::SSL;
        };
        die "Please install IO::Socket::SSL\n" if $@;
        print "\nWorking in SSL Mode\n";
        if ( $certificate) {
          print "SSL parameters will be taken from certificate and public
          key local
      files.\n";
        }
        else {
            print "SSL parameters will be taken from the server during
            handshake.\n";
        }
        $port ||= 9003;
              $socket = IO::Socket::SSL->new (PeerAddr         =>
              $gateway_ip, PeerPort   => $port,
              Proto            => 'tcp', SSL_version      => 'SSLv3',
              SSL_verify_mode  => $certificate,
              SSL_ca_file      => 'Keys\\client-cert.pem',
            );
        } else {
              print "\nWorking in Regular tcp/ip Mode\n";
              $port ||= 9002;
              $socket = IO::Socket::INET->new (PeerAddr => $gateway_ip,
              PeerPort => $port,
              Proto => 'tcp',
              );
  }
  print_params();
  die "Unable to open socket in port $port for server $gateway_ip.
  \nError: $!" unless
$socket;

  my $login = "openconnection user=$user password=$pass\n.\n";
  print "login to the gateway....\n";
```

```
  print $socket "$login";
  sleep 3;

  while ($_=<$socket>) {
        print $_;
      if (/^\.\r/) {last}
  }
  sleep 3;
      print "-------------------------\n\nexecuting the bql....\n";
      print $socket "$bql_statement";
      my $inMessage=0;
      my $message="empty";
      while ($_=<$socket>) {
        print $_;
        if (/<Message type="String">/) {
              $inMessage=1;
        }

        if (m{<Message type="String">(.*)</Message>}) {
              $message.=$1;
              $inMessage=0;
        }
        elsif ($inMessage and m{^(.*)</Message>}) {
              $message.=$1;
              $inMessage=0;
        }
        elsif (/<Message type="String">(.*)/) {
              $message.=$1;
              $inMessage=1;
        }
        elsif ($inMessage) {
              $message.=$_;
        }

        if ($message ne "empty" and !$inMessage) {
              $message =~ s/^empty//g;
              print "$message\n";
              $message="empty";
        }

  if (/^\.\r/) {last}
}

  sleep 3;
  print "-------------------------\n\nlogout from the gateway....\n";
  print $socket "exit\n.\n";
  while ($_=<$socket>) {
  print $_;
  if (/^\.\r/) {last}
  }
}
```

4.2 Running BQL Using Web Services

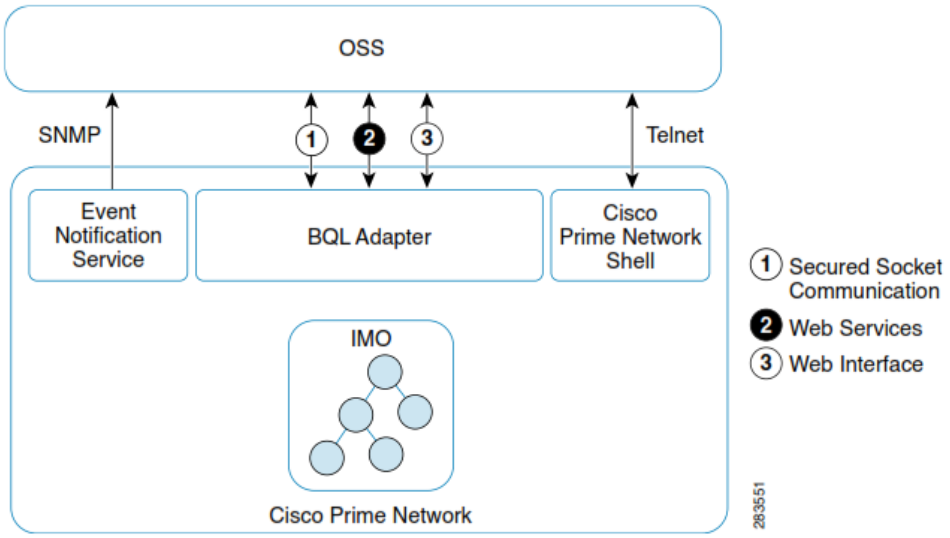
The Cisco Prime Network web services interface is defined using Web Services Definition Language (WSDL) descriptors. Using this interface, you can execute a command and

register for notifications. The key features of the Prime Network web services interface are:

- A web service command executer based on Java API for XML Web Services (JAX-WS). This package is part of Java Standard Edition (JSE), and it is deployed on the new com.sun.net.httpserver.HttpsServer. The command can be executed in both synchronous and asynchronous modes.
- A callback web service destination based on a known schema (WSDL). This callback must be published by the client to receive notifications and asynchronous results. The client sends this callback endpoint reference (EPR) along with the request. The server provides the WSDL for this callback.
- A callback token that the client can send along with the request. In response (with or without notification), the server sends this token with the result. The client can use this mechanism to get multiple responses on the same web services destination.

BQLExecute is the web service request used by the interface. This request has a parameter of type string, which contains the entire BQL request. The web service client must construct this BQL request with the required parameters and XML structure and parse the responses. It must also interpret the returned structure.

Figure 4-5 Cisco Prime Network Web Services Interface



The Prime Network web service is implemented using JAX-WS Framework, and it is interoperable with other web services frameworks such as .NET framework. You can download the WSDL and XML Schema Definition (XSD) from either the Prime Network gateway or the Prime Network Technology Center website.

The client uses SOAP over HTTPS to communicate with the Prime Network gateway web services on port 6081. The server endpoint can handle more than 50 clients simultaneous.

The SOAP header contains the URL of the service, operation name, and authentication information, and the SOAP body contains the BQL payload.
The server endpoint instrumentor contains the following interfaces:

- execute (bql_cid, epr, callbackToken)—Returns the registration result.
- asyncExecute (bql_cid, epr, callbackToken)—Returns the registration ID.
- unregister (registrationId)—Unregisters the registration.

The execute methods (execute and asyncExecute) return a registration ID if the provided retrieval specification (part of the BQL CID) is registered.
The client has to parse the result to understand the server response in terms of registration ID and command result, according to the following table:

Method	Request Type	Response Type
execute	regular	CommandResult
execute	registered	RegistrationResult
asyncExecute	regular	none
asyncExecute	registered	RegistrationId

4.2.1 Prime Network Web Services Endpoint References

A web services endpoint is a reference entity, processor, or resource to which web services messages can be targeted. Endpoint references (EPRs) convey the information needed to identify and reference a web services endpoint, and may be used to:

- Convey the information needed to access a web services endpoint
- Provide addresses for individual messages sent to and from web services

The EPR is a destination address in Uniform Resource Identifier (URI) form and is part of the WS-Addressing specification. It identifies and references a web service endpoint. At a minimum, it contains a URI that locates the endpoint resource (which defines the relationship between a web service and an application), but it can also contain other parameters to make the reference unique.

You can do the following using the Prime Network web services endpoint references:

- GET the service WSDL by sending the request through the URL:

  https://PrimeNetwork_Gateway_IP_Address:6081/ana/ws/executer?wsdl
- Post SOAP requests to the Prime Network gateway web services endpoint and invoke the following web service methods (JAX-WS syntax):
  - String execute (String CID, W3CEndpointReference EPR, String callbackToken)— Executes the provided CID and returns a string representation of the result. The register command returns a notification to the provided EPR along with the registrationId. The callback token

parameter is used by the client to set additional data with the command. This token is attached by the server to the result and to any future notification.

- String asyncExecute (String cid, W3CEndpointReference epr, String callbackToken)—Executes the provided CID in asynchronous mode and returns a string representation of the registrationId (if any). The result is returned to the provided EPR when it is ready. The client call is not blocked until it is ready. The register command returns a notification to the provided EPR along with the registrationId. The callback token parameter is used by the client to set additional data along with the command. This token is attached by the server to the result and to any future notification.
- void unregister (String registrationId)—Used to perform an unregister. The client sends the registrationId to unregister a specific command.

- Create your own W3C endpoint reference (W3CEPR) to get asynchronous results and notifications. The client must implement a remote Callback interface and bind it as a local address to create an endpoint reference.

  The client can adopt either the top-down or bottom-up approach. The server must provide both options to the client:

  - Top-down—The client can use the provided callback WSDL to generate the required Java
    Architecture for XML Binding (JAXB) and JAX-WS artifacts. The implemented callback is based on these generated artifacts. The callback definition schema (WSDL) must be available at http://*PrimeNetwork_Gateway_IP_Address*:6080/ana/ws/callback?wsdl for the callback service.
    The client can use the *wsimport* tool to generate the required artifacts.
  - Bottom-up—The server externalizes the java IRemoteCallback signature. Based on this interface, the client publishes a callback implementation and a local WSDL.

4.2.2 Web Services Prerequisites

Using Prime Network web services requires that:

- Your client operating system supports the standardized web services (WS-*).

  Note For the purposes of this document, we assume you are using a UNIX-based machine. This means that, for example, most of the path names in our code samples use the UNIX or Linux forward slash (/), rather than the Windows backward slash (\).

- You are using a programming language (such as Java, Perl, or Python) that supports web services. You can implement your client application using any

programming language that supports web services; the only requirement is that it must be able to process XML requests and responses.

Note As a convenience, Prime Network provides Java client proxies and JAR files and sample applications based on Java. If you want a client that is not based on Java, you will either have to implement your own client or contact ask-ana@cisco.com for assistance.

- You have experience with web services client-side and server-side programming.
- You have a working knowledge of web services standards, especially Prime Network specifications such as WS-Addressing, WS-Enumeration, WS-Notification, and WS-ResourceFramework (see http://www.w3.org/2002/ws/).
- You have a notification consumer running on an HTTP web server. Notifications are sent via HTTPS. For more information about notification consumers, see http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf.
- The notification consumer has a WS-Resource (or is itself be a WS-Resource) that can process incoming SOAP notification messages.

Web Services Reference URLs

See the following to learn more about web services:

- WS-Addressing: http://www.w3.org/Submission/ws-addressing/
- WS-Enumeration: http://www.w3.org/Submission/WS-Enumeration/
- WS-Notification: http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf
- WS-Resource Framework: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrf
- WS-ResourceLifetime: http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceLifetime-1.2-draft-03.pdf
- WS-ResourceProperties: http://docs.oasis-open.org/wsrf/2004/06/wsrf-WS-ResourceProperties-1.2-draft-04.pdf
- WS-Security: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss
- WSDL: http://www.w3.org/TR/wsdl

4.2.3 Prime Network Web Services Summary

Prime Network supports the following web services:

- *Command Executer*—Runs on the gateway.
- *Remote Callback*—Deployed on the client.

Table 4-1 provides descriptions for the Prime Network web services operations and the corresponding EPRs that are exposed by the Prime Network integration interfaces. Use this table to determine the web service to which your client application should connect. The default port for HTTPS connections to Prime Network gateway is 6081.

Table 4-1 Prime Network Web Services Operations

Web Service	Description
Command Executer	
execute	Executes the BQL commands synchronously.
asyncExecute	Executes the BQL commands asynchronously.
unregister	Unregisters the subscriptions.
Remote Callback Service	
onMessage	Handles all messages (notifications and asynchronous responses) from the Prime Network gateway.

4.2.4 Sample Prime Network Web Services Client

You can download the Prime Network ISDK Java application sample from the Prime Network Technology Center website:

.
Use WebServicesClient, to perform operation using JAX-WS based web services. Running WebServicesClient requires the following:

1. Downloading and Installing the Required **Packages**, page 77
2. Setting up the Java and Eclipse Environment, page 77
3. Updating the Configuration File, page 78
4. Running **WebServicesClient**, page 78

Downloading and Installing the Required Packages
You must download and install the following packages to run the sample web services client:

- Sun Java Development Kit 6.0 from http://java.sun.com/javase/downloads/index.jsp.
- Eclipse 3.x IDE from http://www.eclipse.org/downloads/.
- Prime Network sample web services client from http://developer.cisco.com/web/prime-network.

Follow the instructions in Prime_Network_ISDK_ReadmeFirst.txt to install these packages.

Setting up the Java and Eclipse Environment
Follow the instructions in Cisco_Prime_Network_ISDK_ReadmeFirst.txt to set up the Java and Eclipse environment to run the web services application.

Updating the Configuration File

To update the config.xml file to run WebServicesClient:

1. Navigate to the directory where you have extracted the sample web services client package.
2. Open the config.xml file and update the following parameters:

   gw_server

   - URL: Change the URL to *https://PrimeNetwork_Gateway_IP_Address:PrimeNetwork_Gateway_Port/ana/ws/executer?wsdl*

   Where:

   - *PrimeNetwork_Gateway_IP_Address* = IP address of the Prime Network gateway.
   - *PrimeNetwork_Gateway_Port* = HTTPS (6081) Prime Network gateway port.

   - callbackurl: Change the URL to http://*clientip-address*:*client-port*/my/callback

   Where:

   - *clientip-address* = IP address of your client machine.
   - *client-port* = Port number of your client machine.

   ws_client

   bqlfilepath: Enter the filename of the BQL script file.

   data

   No update is required.

Running WebServicesClient

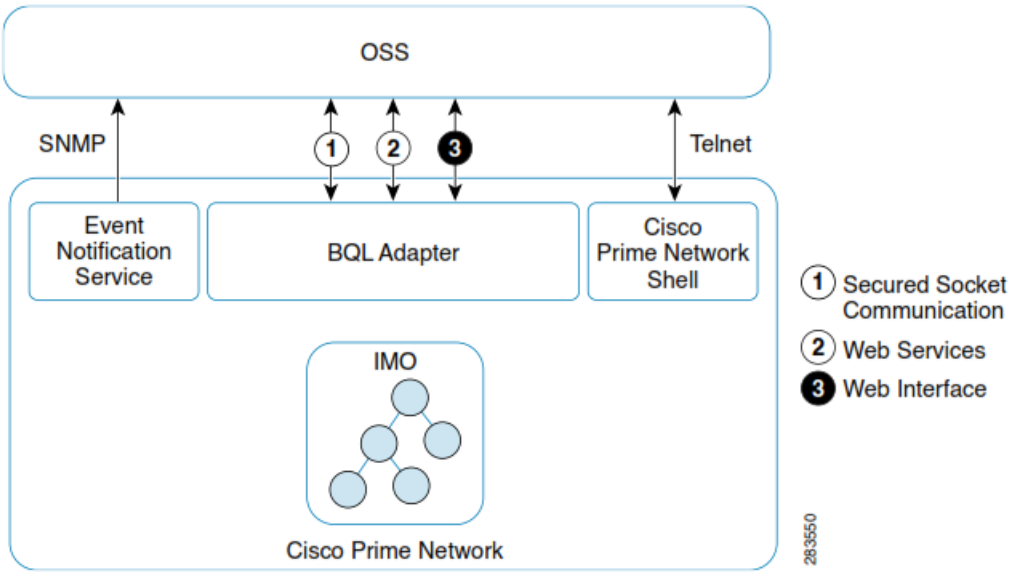
To run WebServicesClient:

1. Launch Eclipse and choose WebServicesClient in the Package Explorer. Under the src folder, choose *WSClient.java*.
2. Right-click and choose **Run As > Run Configurations...** The Run Configurations window appears.
3. In the Arguments tab, enter the Prime Network username and password in the Program Arguments pane. Click **Apply** and **Run**. The BQL response is displayed.

4.3 Running BQL using the Web Interface

You can run BQL commands via a web interface using a standard web browser. This feature is similar to web service, but uses a web interface authentication instead of SOAP envelopes.
Figure 4-6 shows the web interface implementation in Cisco Prime Network.

Figure 4-6 Prime Network Web Interface



The following are the key supported features in the web interface:

- You can have more than 50 simultaneous web interface requests.
- You can use only those commands that have a single response.
- You cannot use commands that include registration.

4.3.1 Sample Prime Network Web Interface Clients

You can download the Prime Network ISDK Java application sample from the Prime Network Technology Center website:

https://developer.cisco.com/site/prime-network/.

Use WebServicesClient to perform operation using JAX-WS based web services. Running WebServicesClient requires the following:

1. Downloading and Installing the Required Packages, page 80
2. Setting up the Java and Eclipse Environment, page 80
3. Updating the Configuration File, page 80

Downloading and Installing the Required Packages

You must download and install the following packages to run the sample web interface clients:

- Sun Java Development Kit 6.0 from http://java.sun.com/javase/downloads/index.jsp.
- Eclipse 3.x IDE from http://archive.eclipse.org/eclipse/downloads.
- Prime Network sample web services client from http://developer.cisco.com/web/prime-network.

Follow the instructions in Prime_Network_ISDK_ReadmeFirst.txt to install these packages.

Setting up the Java and Eclipse Environment

Follow the instructions in Prime_Network_ISDK_ReadmeFirst.txt to set up the Java and Eclipse environment to run the web interface application.

Updating the Configuration File

To update the config.xml file to run web interface clients:

1. Navigate to the directory where you have extracted the sample web interface client package.
2. Open the config.xml file and update the following parameters:

   gw_server

   - Update:
     - *PrimeNetwork_Gateway_IP_Address* =IP address of the Prime Network gateway.
     - *PrimeNetwork_Gateway_Port* =6081 is the HTTPS port number for Prime Network gateway.
   - callbackurl: No update is required.

   ws_client

   Enter the filename of the BQL script file.

   data

   No update is required.

5 BQL Application Examples

This section contains the following topics:

- Best Practices for BQL Parsing
- Processing BQL Notification Messages
- Administering Cisco Prime Network Using BQL
- Retrieving Inventory Data Using BQL
- Generating Standard Reports Using BQL
- Running Command Builder Scripts Using BQL

- Managing Soft Properties Using BQL
- Running Configuration Backup and Restore Operations Using BQL
- Running Compliance Audit Using BQL
- Running Transactions Using BQL
- BQL Error Catalog and Examples

5.1 Best Practices for BQL Parsing

This section describes the best parsing practices when integrating to Cisco Prime Network via the BQL interface. The best practices include:

5.1.1 Use an Existing XML Parser Implementation

Each Prime Network integration is usually written in a programming language, and most, if not all, of these programming languages have existing XML parser implementations. We strongly recommend that you use an existing XML parser rather than implementing a new one.

5.1.2 Do Not Rely on the Order of the Properties Inside the XML Parser

Although it can appear that the order of the properties inside a BQL response is fixed, do not rely on this. XML ISO does not define the order, and therefore the BQL response does not define the order of the different properties. Locating a specific property inside an element should be done by iterating on all of the properties inside an element until the requested property is found.

Note that in this example, the order of the colored keys can be switched.

Example

In different Prime Network versions, two IMO replies can be sent, each containing the same set of properties in a different order.

First IMO reply:

```
<IManagedElement>
  <ID type="Oid">{[ManagedElement(Key=PE-South)]}</ID>
  <CommunicationStateEnum type="Integer">3</CommunicationStateEnum>
  <CpuUsage type="Integer">2</CpuUsage>
  <DRAMFree type="Integer">33141168</DRAMFree>
  <DRAMUsed type="Integer">413264</DRAMUsed>
  <DeviceName type="String">PE-South</DeviceName>
  <DeviceSerialNumber type="String">21293957</DeviceSerialNumber>
…
</IManagedElement>
```

Second IMO reply:

```
<IManagedElement>
  <ID type="Oid">{[ManagedElement(Key=PE-South)]}</ID>
  <CommunicationStateEnum type="Integer">3</CommunicationStateEnum>
  <DRAMFree type="Integer">33141168</DRAMFree>
  <CpuUsage type="Integer">2</CpuUsage>
  <DRAMUsed type="Integer">413264</DRAMUsed>
  <DeviceName type="String">PE-South</DeviceName>
  <DeviceSerialNumber type="String">21293957</DeviceSerialNumber>
…
</IManagedElement>
```

5.1.3 Do Not Count or Validate the Number of Properties

The IMO exposed through BQL can be extended with additional properties across different versions of Prime Network, so validating the number of properties can produce an error. Instead, parse the properties you expect to receive in such a way that any addition does not affect your integration by producing errors.

Example
The following IMO:

```
<IManagedElement>
  <ID type="Oid">{[ManagedElement(Key=PE-South)]}</ID>
  <CommunicationStateEnum type="Integer">3</CommunicationStateEnum>
  <CpuUsage type="Integer">2</CpuUsage>
  <DRAMFree type="Integer">33141168</DRAMFree>
  <DeviceName type="String">PE-South</DeviceName>
  <DeviceSerialNumber type="String">21293957</DeviceSerialNumber>
…
</IManagedElement>
```

can be extended with an additional property:

```
<IManagedElement>
  <ID type="Oid">{[ManagedElement(Key=PE-South)]}</ID>
  <CommunicationStateEnum type="Integer">3</CommunicationStateEnum>
  <DRAMFree type="Integer">33141168</DRAMFree>
  <CpuUsage type="Integer">2</CpuUsage>
  <DRAMUsed type="Integer">413264</DRAMUsed>
  <DeviceName type="String">PE-South</DeviceName>
  <DeviceSerialNumber type="String">21293957</DeviceSerialNumber>
…
</IManagedElement>
```

5.1.4 Do Not Assume the Type of the Property; Parse it from BQL

Property types are declared in the IMO, and might change from Prime Network version to version. Property types should be parsed, and then compared to the expected type to detect any changes.

Example
The following property:

```
<DeviceSerialNumber type="String">21293957</DeviceSerialNumber>
```

can change to:

```
<DeviceSerialNumber type="Integer">21293957</DeviceSerialNumber>
```

5.1.5 Keep the Data Hierarchical Structure

When doing integrations with BQL responses, keep track of the BQL output structure in the integration data. This way, when new hierarchy sons or parents appear, the changes are easier to adjust or disregard.

Example
The following IMO:

```
<IFWComponentContainer>
  <ID type="Oid">{[ManagedElement(Key=PE-
South)][LogicalRoot][FWComponentContainer(Type=1)]}</ID>
  <FWComponents type="IMObjects_Array">
        <IRoutingEntity>
            <ID type="Oid">{[ManagedElement(Key=PE-
      South)][LogicalRoot][FWComponentContainer(Type=1)][RoutingEntity]}</
      ID>
            <Name type="String">Default context</Name>
            <ScriptMetadataOids type="com.sheer.framework.imo.Oid_Array">
        <com.sheer.framework.imo.Oid>{[Command(CommandId=ciscoscriptsrepos
  itory/add- loopback-
  interface)][ScriptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.O
  id>
<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscoscriptsrepository/ad
d-static-route)]
[ScriptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>
<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscoscriptsrepository/sho
w-ip-route)][Sc
riptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>
              </ScriptMetadataOids>
            </IRoutingEntity>
        </FWComponents>
      <ScriptMetadataOids type="">Null</ScriptMetadataOids>
      <TypeEnum type="Integer">1</TypeEnum>
</IFWComponentContainer>
```

might produce a hierarchical structure such as:

```
IFWComponentContainer
  ID
  FWComponents
  IRoutingEntity
  ID Name
  ScriptMetadataOids
        com.sheer.framework.imo.Oid
        com.sheer.framework.imo.Oid
        com.sheer.framework.imo.Oid
```

5.1.6 ID Parsing Should Be Protected from the Addition of Properties

In BQL, each object has an ID that is formatted as in the following example:

```
{[MCNetwork][MCVM(IP=64.103.124.178)][Avm(AvmNumber=600)]}
```

Each value between the square brackets ([ ]) represents an element, while parentheses (( )) contain an element property and a value. When parsing the ID, do not rely on the order or the number of these elements or properties, just on their existence; in other words, if you search for a specific property iteration, search for it on all properties until the property is found, rather than looking for it in a specific place. In this way, if an element or a property is added, the integration is not affected.

Example
If you are searching for the port number in the following object ID:

```
<ID type="Oid">{[ManagedElement(Key=PE-South)][PhysicalRoot][Chassis]
[Module][Port(PortNumber=FastEthernet1/0)]}</ID>
```

and if an additional element named Slot was added to the object ID:

```
<ID type="Oid">{[ManagedElement(Key=PE-South)]
[PhysicalRoot][Chassis][Slot(SlotNum=1)][Module][Port(PortNumber=FastEther
net1/0)]}</ID>
```

The integration code that searches for the PortNumber property can be written in such a way that it will not fail as a result of adding the additional property.

Note For information regarding IMO and API changes from version to version, see the Cisco Prime Network 4.2.2 Release Notes.

5.1.7 Choose the Relevant Data for Retrieval and Registration

When using the fault management BQL commands and Event Notification Service, you must:

- Choose the attributes that are required in the result. Use the Retrieval specification concept. It provides the capability to specify the required and excluded attributes.

  In the following BQL example, only alarm source, state, and severity is retrieved.

```
<key name="requiredProperties">
  <key name="com.sheer.imo.newalarm.IAlarm">
        <entry name="Source"/>
        <entry name="LatestState"/>
        <entry name="SeverityEnum"/>
  </key>
</key>
```

  In the following BQL example, all traps are excluded:

```
<key name="excludedProperties">
  <key name="com.sheer.imo.ITrapValue">
```

```
        <entry name="*"/>
</key>
```

- Plan and choose the relevant data that you want to register for notification. Registering for notification for a large volume of unnecessary data impacts the system performance.

5.1.8 Subscribe Only for Tickets and Ticket Updates

Subscribing only for tickets and ticket updates significantly reduces the data volume that the OSS client needs to handle. In addition, you can reduce notifications by selecting event types and severity.

To avoid receiving a large volume of notifications, do not subscribe for all event categories.

The following SNMP notification example does not include any filters. The notification is sent for all updates. To reduce the volume of data, you can use the filter on tickets, ticket updates, event types, and severity (shown in bold text). If you want to use Exclude filter properties, then *IncludedEventTypes* should have -1 and/or *IncludedSourceIPs* should have 0.0.0.0,0.0.0.0.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Create">
  <param name="imobject">
        <value>
              <IOSSClientInfo>
                <ID type="Oid"> {[OssClientInfoRoot][OSSClientInfo(Id=0)]
                }</ID>
                <Address type="com.sheer.types.IPAddress">64.103.124.249
                </Address>
                <CommunityString type="String">testCommunityString
                </CommunityString>
                <ConnectionType
                type="com.sheer.types.sbc.TransportProtocolEnum">UDP</Conn
                ectionType>
                <Filter type="IOssEventTrapFilter">
                    <ID type="Oid">{[OSSClientFilter(Id=1)]}</ID>
                    <ExcludedEventTypes type="int_Array" />
                    <ExcludedSourceIPs
                    type="com.sheer.types.IPSubnet_Array" />
                    <IncludedEventCategories
                    type="com.sheer.types.enums.NotificationCategoriesEnum
                    _Array">

<com.sheer.types.enums.NotificationCategoriesEnum>Provisioning</com.sheer.t
ypes.enums.NotificationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>Security</com.sheer.types
.enums.NotificationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>Service</com.sheer.types.
enums.NotificationCategoriesEnum>
```

```
<com.sheer.types.enums.NotificationCategoriesEnum>Syslog</com.sheer.types.e
nums.NotificationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>System</com.sheer.types.e
nums.NotificationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>Ticket</com.sheer.types.e
nums.NotificationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>TicketUpdate</com.sheer.t
ypes.enums.Noti ficationCategoriesEnum>
<com.sheer.types.enums.NotificationCategoriesEnum>V1Trap</com.sheer.types.
enums.Notificati
onCategoriesEnum>
<com.sheer.types.enums.NotificationCategoriesEnum>V2Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V3Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>
  </IncludedEventCategories>
  <IncludedEventTypes type="int_Array">
        <int>-1</int>
  </IncludedEventTypes>
  <IncludedSeverities type="int_Array">
      <int>0</int>
      <int>1</int>
      <int>2</int>
      <int>3</int>
      <int>4</int>
      <int>5</int>
      <int>6</int>
  </IncludedSeverities>
  <IncludedSourceIPs type="com.sheer.types.IPSubnet_Array">

<com.sheer.types.IPSubnet>0.0.0.0,0.0.0.0</com.sheer.types.IPSubnet>
        </IncludedSourceIPs>
      </Filter>
      <Port type="Integer">162</Port>
      <SnmpVersion
  type="com.sheer.types.enums.NotificationSnmpVersionEnum">SnmpV2</SnmpVer
  sion>
      <TypeEnum
  type="com.sheer.types.enums.NotificationTypeEnum">EventTrapNotification<
  /TypeEnum>
      </IOSSClientInfo>
    </value>
  </param>
</command>
.
```

5.2 Processing BQL Notification Messages

When a client application collects information such as inventory data from Cisco Prime Network, the results might be affected if the status of a device is up or down. To keep track of network device changes, a client application might poll the entire inventory periodically and determine each time if the device status has changed. However, if the

time between calls is very small, the amount of network traffic and CPU use increases because a complete inventory is requested each time. Or, there might be dozens of applications all waiting for an event to happen.

Instead, using the notification approach, the same client application can subscribe to Prime Network notifications to receive updates as they happen. This approach is a more efficient and real-time way for the client application to keep its information synchronized with Prime Network.

Another example of using notifications is when correlated alerts are reported by Prime Network and a trouble ticket client application needs to open tickets. The client application can first subscribe to the types of Prime Network notifications it needs, wait to receive notifications as they happen, and then create or update the trouble ticket for that alert.

Prime Network supports a *register and unregister* XML-based notification mechanism, where:

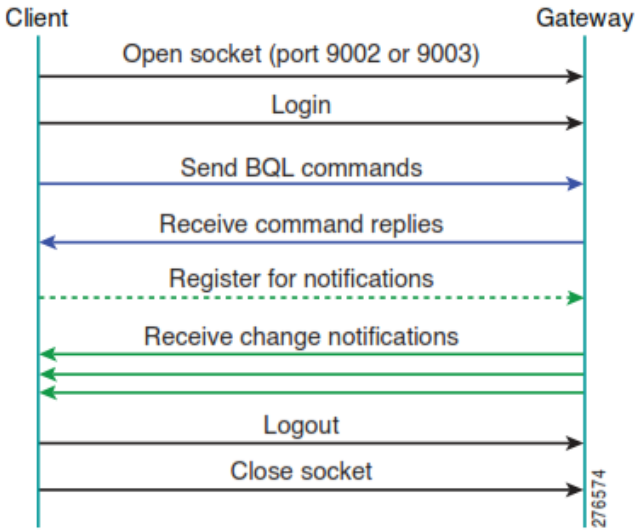
- The Prime Network gateway broadcasts notifications.
- The client applications register only to those notifications in which they are interested.

Based on your registration, notification messages are sent asynchronously. These messages are sent by the Prime Network gateway and can be originated in either the Prime Network gateway or unit.

Note Notification messages are sent in the same socket that is used for registration. If the socket is closed, the registration is unregistered automatically in the Prime Network gateway. It is also sent to the JAX-WS client application if the application is running (for example, on the Eclipse console).

Figure 5-1 shows the sequence of interaction between client applications and the Prime Network gateway over a secured socket connection.

Figure 5-1 Interaction between Client Applications and the Prime Network Gateway



The notification object sent by Prime Network is an IMO describing a change in another IMO. This is always of type IMObject_Array, in which each element represents a single property change. The changed object contains the following information:

- Object identifier
- Property name
- New property value
- Sequence and time stamp (these objects are displayed based on the implementation)

The following are the various types of notification messages supported by Prime Network:

- IScalarNotification—Describes the change in a property for an existing object; for example, port status. See IScalarNotification, page 93 for an example.
- IAddNotification—Notifies you about a new object that has been added to the system; for example, a new alarm ticket. See IAddNotification, page 94 for an example.
- IRemoveNotification—Sent by a containing object to notify you that one of its internal (nested) objects has been removed; for example, a card removed from a shelf. See IRemoveNotification, page 101 for an example.
- IObjectDeleteNotification—Sent by an object which has been deleted; for example, a route entry. See IObjectDeleteNotification, page 102 for an example.

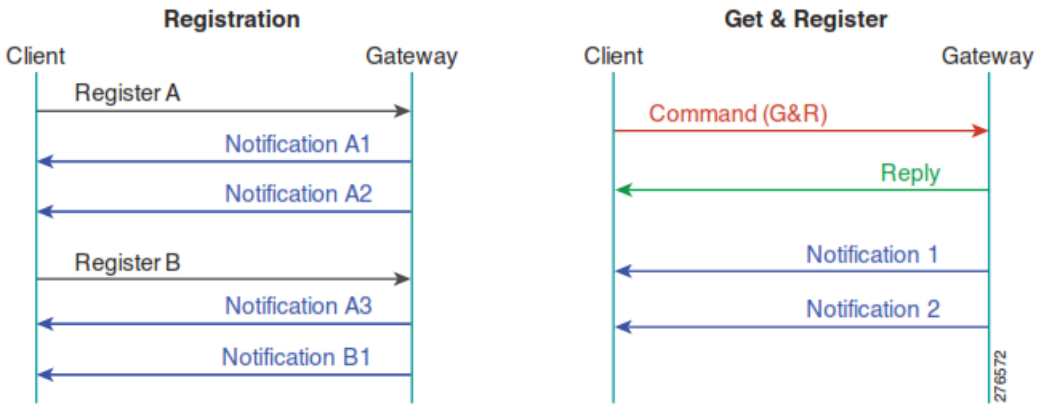
Note IRemoveNotification and IObjectDeleteNotification are not the same. See IRemoveNotification, page 101 and IObjectDeleteNotification, page 102.

5.2.1 Registering for BQL Notification Service

If the register flag in the RS is set to true, the system returns the IMO construct that matches this RS and registers for changes of the data. Any change detected in any of the data items is sent in an unsolicited message to the BQL client.
At the time of the registration, if you want to retrieve the data and subscribe for notification, then you must use the **Get** and **Register** commands. They retrieve the data and subscribe for notification. For example, see Registering for Notification Using the BQL Get and Register Command, page 104.

If you want to subscribe only for the changes without retrieving the data, then you must use the **Register** command. For example, see
Registering for Notification Using the BQL Register **Command**, page 109.

Figure 5-2 BQL Register Command and BQL Get Command



When a BQL session is closed, the system clears all registrations associated with that session. Therefore, you do not need to explicitly unregister before closing the BQL session.

You can also have multiple registrations. When registering for notifications, you can optionally specify a registration ID for each registration. This registration ID can be used later for unregistering. The registration ID should be unique; it is up to the client application to ensure uniqueness. If you perform two registrations with the same ID, the second registration overrides the first. The first registration remains active and sends notifications, but you cannot unregister it. In such situations, the only way to unregister the first registration is by issuing an **unregister all** command.

To specify a registration ID in a register command, append commandId=ID to the beginning of the **Get** command; for example:

```
commandId=1
<?xml version="1.0" encoding="UTF-8"?>
<command name="Get">
<param name="oid">
<value>{[WorkflowManagement][Workflow(Id=5)]}</value>
</param>
<param name="rs">
<value><key name="com.sheer.imo.keys.IWorkflowManagementOid">
<entry name="depth">10</entry>
<entry name="register">true</entry>
<entry name="cachedResultAcceptable">false</entry>
<key name="requiredProperties">
<key name="com.sheer.imo.IWorkflow">
<entry name="StateEnum"/>
<entry name="Template"/>
</key>
</key>
</key></value>
</param>
</command>
.
```

In the notification message that you receive (see Understanding Notification Messages, page 92), the first line contains the information on the commandId whenever commandId is prefixed while using the **Get** and **Register** or **Register** command. This helps the client applications to parse and forward the notifications to the appropriate internal application. In other words, the alarm ticket notifications are forwarded to the Trouble Ticketing application; inventory-related notifications are forwarded to the Discovery application.

The notification messages have a different EOT sequence. The EOT sequence "\n$\n" instead of "\n.\n" indicates to the Prime Network client that more information may be expected. The BQL client can issue multiple concurrent registrations that are identified by name.

See Registering for Notification for Specific Events, page 24 for information on registering for notification for an event type.

5.2.2 Parsing a Notification Message

When the client receives any notification message, a callback method is called.

For further processing, the notification parsing module should be called from this method. This parsing method should:

1. Retrieve the sequence number and time stamp from the message.
2. Retrieve the message content and check the type of message to determine whether it is IAddNotification, IScalarNotification, IRemoveNotification, or IObjectDeleteNotification.
3. Retrieve the property name of the object.
4. Retrieve the new property value.
5. Parse and persist the new property value.

5.2.2.1 Understanding Notification Messages

A notification is an IMO that describes a change in another IMO.
See the example in Registering for Notification Using the BQL Get and Register Command, page 104, to register a map for notification. The following example displays the notification message that is sent to the client application when the map name is changed:

```
commandId=111
<?xml version="1.0" encoding="UTF-8"?>
<IMObjects_Array>
  <IScalarNotification type="IScalarNotification" instance_id="1">
        <ID type="Oid">{[Notification(Source={[HierarchyNode(Id=2204)]})]}
        </ID>
        <NewIMO type="IHierarchyNode" instance_id="2">
              <ID type="Oid">{[HierarchyNode(Id=2204)]}</ID>
              <Name type="String">CiscoMap1</Name>
        </NewIMO>
        <OldIMO type="IHierarchyNode" instance_id="3">
```

```
            <ID type="Oid">{[HierarchyNode(Id=2204)]}</ID>
            <Name type="String">CiscoMap</Name>
        </OldIMO>
        <PropertyName type="String">Name</PropertyName>
    </IScalarNotification>
    <IScalarNotification type="IScalarNotification" instance_id="4">
        <NewIMO type="IMap" instance_id="5">
              <ID type="Oid">{[Map(Id=58003)]}</ID>
              <HierarchyRootOid type="Oid">{[HierarchyNode(Id=2204)]}
              </HierarchyRootOid>
              <Name type="String">CiscoMap1</Name>
        </NewIMO>
        <PropertyName type="String">Name</PropertyName>
</IScalarNotification>
</IMObjects_Array>
```

In this example, the notification indicates that in the IMO of type IHierarchyNode, the map name has been updated from CiscoMap to CiscoMap1. The notification is always of type IMObjects_Array and contains an IMO for each notification. In this example, it contains only a single notification of type IScalarNotification, which indicates a change in a single property. The notification includes the command ID and, for each object, the changed property type, new value, and old value.

The following sections describe:

- IScalarNotification, page 93
- IAddNotification, page 94
- IRemoveNotification, page 101
- IObjectDeleteNotification, page 102

IScalarNotification

IScalarNotification describes the change in a property for an IMO. The following example contains the notification message when the map name is updated:

```
commandId=111
<?xml version="1.0" encoding="UTF-8"?>
<IMObjects_Array>
  <IScalarNotification type="IScalarNotification" instance_id="1">
        <ID type="Oid">{[Notification(Source={[HierarchyNode(Id=2204)]})]}
        </ID>
        <NewIMO type="IHierarchyNode" instance_id="2">
              <ID type="Oid">{[HierarchyNode(Id=2204)]}</ID>
              <Name type="String">CiscoMap1</Name>
        </NewIMO>
        <OldIMO type="IHierarchyNode" instance_id="3">
              <ID type="Oid">{[HierarchyNode(Id=2204)]}</ID>
              <Name type="String">CiscoMap</Name>
        </OldIMO>
      <PropertyName type="String">Name</PropertyName>
  </IScalarNotification>
  <IScalarNotification type="IScalarNotification" instance_id="4">
        <NewIMO type="IMap" instance_id="5">
              <ID type="Oid">{[Map(Id=58003)]}</ID>
              <HierarchyRootOid
              type="Oid">{[HierarchyNode(Id=2204)]}</HierarchyRootOid>
```

```
            <Name type="String">CiscoMap1</Name>
        </NewIMO>
        <PropertyName type="String">Name</PropertyName>
  </IScalarNotification>
</IMObjects_Array>
```

IScalarNotification has the following elements:

Element	Description
ID	The OID of the notification. In this example, the OID is IHierarchyNode.
NewIMO	The IMO after the change. In this example, the new IMO value is CiscoMap1.
OldIMO	The IMO before the change. In this example, the old IMO value is CiscoMap. This element is not displayed for all IMOs; it is displayed based on the implementation.
PropertyName	The property in the IMO that was changed. In this example, the

IAddNotification

IAddNotification indicates that an IMO has been added to the system. The following example contains the notification message when a new network element is added to the map:

```
commandId=111
<?xml version="1.0" encoding="UTF-8"?>
<IMObjects_Array>
  <IAddNotification type="IAddNotification" instance_id="1">
    <ID
type="Oid">{[Notification(Source={[HierarchyNode(Id=2204)]})]}</ID>
    <NewIMO type="IHierarchyNode" instance_id="2">
      <ID type="Oid">{[HierarchyNode(Id=2204)]}</ID>
      <Aspects type="IMObjects_Array">
        <IHierarchyChildrenAspect type="IHierarchyChildrenAspect"
instance_id="3">
          <ID
type="Oid">{[HierarchyNode(Id=2204)][HierarchyChildrenAspect]}</ID>
          <Children type="IMObjects_Array">
            <IHierarchyNode type="IHierarchyNode" instance_id="4">
              <ID type="Oid">{[HierarchyNode(Id=2208)]}</ID>
              <Aspects type="IMObjects_Array">
                <IContainedImo type="IContainedImo" instance_id="5">
                  <ID
type="Oid">{[HierarchyNode(Id=2208)][ContainedImo]}</ID>
                  <Imo type="IManagedElement" instance_id="6">
                    <ID
type="Oid">{[ManagedElement(Key=10.77.212.205)]}</ID>
                    <Aspects type="IMObjects_Array">
                      <ISeverityAspect type="ISeverityAspect"
instance_id="7">
```

```
                        <ID
type="Oid">{[ManagedElement(Key=10.77.212.205)][SeverityAspect]}</ID>
                        <ClearedNum type="Integer">0</ClearedNum>
                        <CriticalNum type="Integer">0</CriticalNum>
                        <IndeterminateNum
type="Integer">0</IndeterminateNum>
                        <InfoNum type="Integer">0</InfoNum>
                        <MajorNum type="Integer">1</MajorNum>
                        <MinorNum type="Integer">0</MinorNum>
                        <NotAckedClearedNum
type="Integer">0</NotAckedClearedNum>
                        <NotAckedCriticalNum
type="Integer">0</NotAckedCriticalNum>
                        <NotAckedIndeterminateNum
type="Integer">0</NotAckedIndeterminateNum>
                        <NotAckedInfoNum
type="Integer">0</NotAckedInfoNum>
                        <NotAckedMajorNum
type="Integer">1</NotAckedMajorNum>
                        <NotAckedMinorNum
type="Integer">0</NotAckedMinorNum>
                        <NotAckedPropClearedNum
type="Integer">17</NotAckedPropClearedNum>
                        <NotAckedPropCriticalNum
type="Integer">0</NotAckedPropCriticalNum>
                        <NotAckedPropIndeterminateNum
type="Integer">0</NotAckedPropIndeterminateNum>
                        <NotAckedPropInfoNum
type="Integer">11</NotAckedPropInfoNum>
                        <NotAckedPropMajorNum
type="Integer">5</NotAckedPropMajorNum>
                        <NotAckedPropMinorNum
type="Integer">7</NotAckedPropMinorNum>
                        <NotAckedPropWarningNum
type="Integer">0</NotAckedPropWarningNum>
                        <NotAckedWarningNum
type="Integer">0</NotAckedWarningNum>
                        <PropClearedNum type="Integer">17</PropClearedNum>
                        <PropCriticalNum
type="Integer">0</PropCriticalNum>
                        <PropIndeterminateNum
type="Integer">0</PropIndeterminateNum>
                        <PropInfoNum type="Integer">11</PropInfoNum>
                        <PropMajorNum type="Integer">5</PropMajorNum>
                        <PropMinorNum type="Integer">7</PropMinorNum>
                        <PropWarningNum type="Integer">0</PropWarningNum>
                        <WarningNum type="Integer">0</WarningNum>
                      </ISeverityAspect>
                      <newalarm.ITicketListAspect
type=="newalarm.ITicketListAspect"
instance_id="8">
                        <ID
type="Oid">{[ManagedElement(Key=10.77.212.205)][TicketListAspect]}</ID>
                        <Tickets type="IMObjects_Array">
                          <newalarm.ITicket type="newalarm.ITicket"
instance_id="9">
                            <ID type="Oid">{[NewAlarm(Id=50)]}</ID>
```

```
                              <AffectedDevicesCount
type="Integer">1</AffectedDevicesCount>
                              <AggregatedAckStateEnum
type="Integer">0</AggregatedAckStateEnum>
                              <AggregatedSeverityEnum
type="Integer">5</AggregatedSeverityEnum>
                              <AlarmCount type="Integer">1</AlarmCount>
                              <Archived type="Boolean">false</Archived>
                              <AutoCleared
type="Boolean">false</AutoCleared>
                              <DuplicationCount
type="Integer">1</DuplicationCount>
                              <EventCount type="Integer">1</EventCount>
                              <LastModificationTime
type="java.util.Date">Tue Jan 19
19:07:50 IST 2010</LastModificationTime>
                              <LatestState type="String">cefc FRU
removed</LatestState>
                              <ReductionCount
type="Integer">1</ReductionCount>
                              <Source
type="Oid">{[ManagedElement(Key=10.77.212.205)][Trap][ServiceEvent(DiffObj
ect=11)]}</Sourc
e>
                           </newalarm.ITicket>
                           <newalarm.ITicket type="newalarm.ITicket"
instance_id="10">
                              <ID type="Oid">{[NewAlarm(Id=101)]}</ID>
                              <AffectedDevicesCount
type="Integer">1</AffectedDevicesCount>
                              <AggregatedAckStateEnum
type="Integer">0</AggregatedAckStateEnum>
                              <AggregatedSeverityEnum
type="Integer">4</AggregatedSeverityEnum>
                              <AlarmCount type="Integer">1</AlarmCount>
                              <Archived type="Boolean">false</Archived>
                              <AutoCleared
type="Boolean">false</AutoCleared>
                              <DuplicationCount
type="Integer">1</DuplicationCount>
                              <EventCount type="Integer">1</EventCount>
                              <LastModificationTime
type="java.util.Date">Wed Jan 20
16:54:48 IST 2010</LastModificationTime>
                              <LatestState type="String">keepalive not
configured</LatestState>
                              <ReductionCount
type="Integer">1</ReductionCount>
                              <Source
type="Oid">{[ManagedElement(Key=10.77.212.205)][LogicalRoot][Context(Conte
xtName=Default
context)][TunnelContainer(TunnelType=4)][TunnelGre(TunnelName=Tunnel192)]}
</Source>
                           </newalarm.ITicket>
                           <newalarm.ITicket>
                              <ID type="Oid">{[NewAlarm(Id=191)]}</ID>
```

```
                              <AffectedDevicesCount
type="Integer">1</AffectedDevicesCount>
                              <AggregatedAckStateEnum
type="Integer">0</AggregatedAckStateEnum>
                              <AggregatedSeverityEnum
type="Integer">5</AggregatedSeverityEnum>
                              <AlarmCount type="Integer">2</AlarmCount>
                              <Archived type="Boolean">false</Archived>
                              <AutoCleared
type="Boolean">false</AutoCleared>
                              <DuplicationCount
type="Integer">2</DuplicationCount>
                              <EventCount type="Integer">3</EventCount>
                              <LastModificationTime
type="java.util.Date">Sat Jan 23
03:08:43 IST 2010</LastModificationTime>
                              <LatestState type="String">Card
in</LatestState>
                              <ReductionCount
type="Integer">3</ReductionCount>
                              <Source
type="Oid">{[ManagedElement(Key=10.77.212.205)][PhysicalRoot][Chassis][Slo
t(SlotNum=0)][Mo
dule][Slot(SlotNum=4)][Module]}</Source>
                            </newalarm.ITicket>
                            <newalarm.ITicket>
                              <ID type="Oid">{[NewAlarm(Id=107)]}</ID>
                              <AffectedDevicesCount
type="Integer">1</AffectedDevicesCount>
                              <AggregatedAckStateEnum
type="Integer">0</AggregatedAckStateEnum>
                              <AggregatedSeverityEnum
type="Integer">4</AggregatedSeverityEnum>
                              <AlarmCount type="Integer">1</AlarmCount>
                              <Archived type="Boolean">false</Archived>
                              <AutoCleared
type="Boolean">false</AutoCleared>
                              <DuplicationCount
type="Integer">2</DuplicationCount>
                              <EventCount type="Integer">2</EventCount>
                              <LastModificationTime
type="java.util.Date">Thu Jan 21
23:45:48 IST 2010</LastModificationTime>
                              <LatestState type="String">keepalive not
configured</LatestState>
                              <ReductionCount
type="Integer">2</ReductionCount>
                              <Source
type="Oid">{[ManagedElement(Key=10.77.212.205)][LogicalRoot][Context(Conte
xtName=Default
context)][TunnelContainer(TunnelType=4)][TunnelGre(TunnelName=Tunnel192)]}
</Source>
                            </newalarm.ITicket>
                            <newalarm.ITicket>
                              <ID type="Oid">{[NewAlarm(Id=97)]}</ID>
                              <AffectedDevicesCount
type="Integer">1</AffectedDevicesCount>
```

```
                              <AggregatedAckStateEnum
type="Integer">0</AggregatedAckStateEnum>
                              <AggregatedSeverityEnum
type="Integer">5</AggregatedSeverityEnum>
                              <AlarmCount type="Integer">1</AlarmCount>
                              <Archived type="Boolean">false</Archived>
                              <AutoCleared
type="Boolean">false</AutoCleared>
                              <DuplicationCount
type="Integer">3</DuplicationCount>
                              <EventCount type="Integer">3</EventCount>
                              <LastModificationTime
type="java.util.Date">Thu Jan 21
02:40:56 IST 2010</LastModificationTime>
                              <LatestState type="String">Device
unreachable</LatestState>
                              <ReductionCount
type="Integer">3</ReductionCount>
                              <Source
type="Oid">{[ManagedElement(Key=10.77.212.205)]}</Source>
                            </newalarm.ITicket>
                            <newalarm.ITicket>
                              <ID type="Oid">{[NewAlarm(Id=193)]}</ID>
                              <AffectedDevicesCount
type="Integer">1</AffectedDevicesCount>
                              <AggregatedAckStateEnum
type="Integer">0</AggregatedAckStateEnum>
                              <AggregatedSeverityEnum
type="Integer">5</AggregatedSeverityEnum>
                              <AlarmCount type="Integer">19</AlarmCount>
                              <Archived type="Boolean">false</Archived>
                              <AutoCleared
type="Boolean">false</AutoCleared>
                              <DuplicationCount
type="Integer">4</DuplicationCount>
                              <EventCount type="Integer">49</EventCount>
                              <LastModificationTime
type="java.util.Date">Fri Jan 22
04:44:13 IST 2010</LastModificationTime>
                              <LatestState type="String">Port
up</LatestState>
                              <ReductionCount
type="Integer">49</ReductionCount>
                              <Source
type="Oid">{[ManagedElement(Key=10.77.212.205)][PhysicalRoot][Chassis][Slo
t(SlotNum=0)][Mo
dule][Port(PortNumber=GigabitEthernet0/3)][PhysicalLayer]}</Source>
                            </newalarm.ITicket>
                          </Tickets>
                        </newalarm.ITicketListAspect>
                      </Aspects>
                      <CommunicationStateEnum
type="Integer">3</CommunicationStateEnum>
                      <DeviceName type="String">10.77.212.205</DeviceName>
                      <ElementCategoryEnum
type="Integer">4</ElementCategoryEnum>
                      <ElementType type="String">Cisco 7206VXR</ElementType>
```

```
                    <ElementTypeKey
type="String">CISCO_7206VXR</ElementTypeKey>
                    <IP
type="com.sheer.types.IPAddress">10.77.212.205</IP>
                    <InvestigationStateEnum
type="Integer">11</InvestigationStateEnum>
                    <SoftwareVersion
type="String">12.4(20)T4</SoftwareVersion>
                    <SysDescription type="String">Cisco IOS Software, 7200
Software
(C7200P-ADVENTERPRISEK9-M), Version 12.4(20)T4, RELEASE SOFTWARE (fc4)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2009 by Cisco Systems, Inc.
Compiled Wed 02-Sep-09 01:54 by prod_rel_team</SysDescription>
                    <SysLocation type="String" />
                    <SysName type="String">PE-7200</SysName>
                    <SysUpTime type="java.util.Date">Fri Jan 22 16:39:11
IST
2010</SysUpTime>
                    <VendorEnum type="Integer">3</VendorEnum>
                  </Imo>
                </IContainedImo>
              </Aspects>
              <ContainedObjectOid
type="Oid">{[ManagedElement(Key=10.77.212.205)]}</ContainedObjectOid>
              <Leaf type="Boolean">true</Leaf>
              <ManagedParent type="">Null</ManagedParent>
              <Map type="Oid">{[Map(Id=58003)]}</Map>
              <Name type="String">10.77.212.205</Name>
            </IHierarchyNode>
          </Children>
        </IHierarchyChildrenAspect>
      </Aspects>
      <Children type="com.sheer.framework.imo.Oid_Array">

<com.sheer.framework.imo.Oid>{[HierarchyNode(Id=2208)]}</com.sheer.framewo
rk.imo.Oid>
      </Children>
    </NewIMO>
    <PropertyName type="String">Children</PropertyName>
  </IAddNotification>
  <IAddNotification type="IAddNotification" instance_id="11">
    <ID
type="Oid">{[Map(Id=58003)][Notification(SequenceNumber=6)(Source={[Map(Id
=58003)]})(Time=
1264498501454)]}</ID>
    <NewIMO type="IMap" instance_id="12">
      <ID type="Oid">{[Map(Id=58003)]}</ID>
      <Aspects type="IMObjects_Array">
        <IMapLinksAspect type="IMapLinksAspect" instance_id="13">
          <ID type="Oid">{[Map(Id=58003)][MapLinksAspect]}</ID>
          <Links type="IMObjects_Array">
            <ITopologicalLink type="ITopologicalLink" instance_id="14">
              <ID
type="Oid">{[TopologicalLink(AEndPoint={[ManagedElement(Key=10.77.212.205)
][LogicalRoot][F
```

```
WComponentContainer(Type=1)][RoutingEntity][IpInterface(IpInterfaceName=Gi
gabitEthernet0/3
)][Mpls]})(LinkType=6)(TunnelID=-
1)(ZEndPoint={[ManagedElement(Key=ASR1004)][LogicalRoot][
FWComponentContainer(Type=1)][RoutingEntity][IpInterface(IpInterfaceName=G
igabitEthernet1/
3/1)][Mpls]})]}</ID>
              <BiDirectional type="Boolean">true</BiDirectional>
              <ConnectionInformation
type="String">IP</ConnectionInformation>
              <DetectionTypeEnum type="Integer">2</DetectionTypeEnum>
              <LinkTypeEnum type="Integer">6</LinkTypeEnum>
              <MaintenanceMode type="Boolean">false</MaintenanceMode>
            </ITopologicalLink>
            <ITopologicalLink type="ITopologicalLink" instance_id="15">
              <ID
type="Oid">{[TopologicalLink(AEndPoint={[ManagedElement(Key=10.77.212.205)
][LogicalRoot][F
WComponentContainer(Type=1)][RoutingEntity][IpInterface(IpInterfaceName=Gi
gabitEthernet0/3
)][Mpls]})(LinkType=6)(TunnelID=-
1)(ZEndPoint={[ManagedElement(Key=CiscoGSRXR)][LogicalRoo
t][FWComponentContainer(Type=1)][RoutingEntity][IpInterface(IpInterfaceNam
e=GigabitEtherne
t0/2/0/0)][Mpls]})]}</ID>
              <BiDirectional type="Boolean">true</BiDirectional>
              <ConnectionInformation
type="String">IP</ConnectionInformation>
              <DetectionTypeEnum type="Integer">2</DetectionTypeEnum>
              <LinkTypeEnum type="Integer">6</LinkTypeEnum>
              <MaintenanceMode type="Boolean">false</MaintenanceMode>
            </ITopologicalLink>
          </Links>
        </IMapLinksAspect>
      </Aspects>
      <Links type="com.sheer.framework.imo.Oid_Array">

<com.sheer.framework.imo.Oid>{[TopologicalLink(AEndPoint={[ManagedElement(
Key=10.77.212.20
5)][LogicalRoot][FWComponentContainer(Type=1)][RoutingEntity][IpInterface(
IpInterfaceName=
GigabitEthernet0/3)][Mpls]})(LinkType=6)(TunnelID=-
1)(ZEndPoint={[ManagedElement(Key=ASR10
04)][LogicalRoot][FWComponentContainer(Type=1)][RoutingEntity][IpInterface
(IpInterfaceName
=GigabitEthernet1/3/1)][Mpls]})]}</com.sheer.framework.imo.Oid>

<com.sheer.framework.imo.Oid>{[TopologicalLink(AEndPoint={[ManagedElement(
Key=10.77.212.20
5)][LogicalRoot][FWComponentContainer(Type=1)][RoutingEntity][IpInterface(
IpInterfaceName=
GigabitEthernet0/3)][Mpls]})(LinkType=6)(TunnelID=-
1)(ZEndPoint={[ManagedElement(Key=Cisco
GSRXR)][LogicalRoot][FWComponentContainer(Type=1)][RoutingEntity][IpInterf
ace(IpInterfaceN
ame=GigabitEthernet0/2/0/0)][Mpls]})]}</com.sheer.framework.imo.Oid>
      </Links>
```

```
    </NewIMO>
    <PropertyName type="String">Links</PropertyName>
  </IAddNotification>
</IMObjects_Array>
```

IAddNotification has the following elements:

Element	Description
ID	The OID of the notification. In this example, the OID is IHierarchyNode.
NewIMO	The added IMO. In this example, the newly added IMO is an IManagedElement, 10.77.212.205.
PropertyName	The table whose element was added.

IRemoveNotification

IRemoveNotification is sent by a containing object to indicate that one of its internal IMOs has been deleted. The following example contains the notification message when a network element is deleted from the map:

```
commandId=111
<?xml version="1.0" encoding="UTF-8"?>
<IMObjects_Array>
  <IRemoveNotification type="IRemoveNotification" instance_id="1">
      <ID
      type="Oid">{[Notification(Source={[HierarchyNode(Id=2204)]})]}</ID>
      <NewIMO type="IHierarchyNode" instance_id="2">
        <ID type="Oid">{[HierarchyNode(Id=2204)]}</ID>
        <Children type="com.sheer.framework.imo.Oid_Array">

<com.sheer.framework.imo.Oid>{[HierarchyNode(Id=2208)]}</com.sheer.framewor
k.imo.Oid>
        </Children>
      </NewIMO>
      <PropertyName type="String">Children</PropertyName>
    </IRemoveNotification>
    <IRemoveNotification type="IRemoveNotification" instance_id="3">
        <ID
        type="Oid">{[Map(Id=58003)][Notification(SequenceNumber=7)(Source=
  {[Map(Id=58003)]})(Time=
1264499375771)]}</ID>
              <NewIMO type="IMap" instance_id="4">
              <ID type="Oid">{[Map(Id=58003)]}</ID>
              <Links type="com.sheer.framework.imo.Oid_Array">

<com.sheer.framework.imo.Oid>{[TopologicalLink(AEndPoint={[ManagedElement(K
ey=10.77.212.20
5)][LogicalRoot][FWComponentContainer(Type=1)][RoutingEntity][IpInterface(I
pInterfaceName= GigabitEthernet0/3)][Mpls]})(LinkType=6)(TunnelID=-
1)(ZEndPoint={[ManagedElement(Key=ASR10
04)][LogicalRoot][FWComponentContainer(Type=1)][RoutingEntity][IpInterface(
IpInterfaceName
=GigabitEthernet1/3/1)][Mpls]})]}</com.sheer.framework.imo.Oid>
```

```
<com.sheer.framework.imo.Oid>{[TopologicalLink(AEndPoint={[ManagedElement(K
ey=10.77.212.20
5)][LogicalRoot][FWComponentContainer(Type=1)][RoutingEntity][IpInterface(I
pInterfaceName= GigabitEthernet0/3)][Mpls]})(LinkType=6)(TunnelID=-
1)(ZEndPoint={[ManagedElement(Key=Cisco
GSRXR)][LogicalRoot][FWComponentContainer(Type=1)][RoutingEntity][IpInterfa
ce(IpInterfaceN
ame=GigabitEthernet0/2/0/0)][Mpls]})]}</com.sheer.framework.imo.Oid>
            </Links>
        </NewIMO>
      <PropertyName type="String">Links</PropertyName>
  </IRemoveNotification>
</IMObjects_Array>
```

IRemoveNotification has the following elements:

Element	Description
ID	The OID of the notification. In this example, the OID is IHierarchyNode.
NewIMO	The OID of the removed element is always given under the NewIMO key. In this example, the newly added IMO is an IManagedElement,
PropertyName	The containing object whose element was deleted.

IObjectDeleteNotification

IObjectDeleteNotification is sent by an object when it is deleted. The following example contains the notification message when a map (map ID 58003) is deleted from the Prime Network Vision:

```
commandId=111
<?xml version="1.0" encoding="UTF-8"?>
<IMObjects_Array>
  <IRemoveNotification type="IRemoveNotification" instance_id="1">
        <ID type="Oid">{[Notification(Source={[Map(Id=58003)]})]}</ID>
        <NewIMO type="IMO" instance_id="2">
              <ID type="Oid">{[Map(Id=58003)]}</ID>
              <Aspects type="IMObjects_Array">
                    <IAspect type="IAspect" instance_id="3">
                          <ID
                      type="Oid">{[Map(Id=58003)][MapDataAspect]}</ID>
                    </IAspect>
              </Aspects>
        </NewIMO>
        <PropertyName type="String">Aspects</PropertyName>
    </IRemoveNotification>
  <IObjectDeleteNotification type="IObjectDeleteNotification"
instance_id="4">
        <Oid type="Oid">{[Map(Id=58003)]}</Oid>
  </IObjectDeleteNotification>
</IMObjects_Array>
```

IObjectDeletedNotification reports the OID of the deleted element.

Note IRemoveNotification is sent whenever any child object is deleted. IObjectDeleteNotification is sent whenever the object is deleted.

IRemoveNotification is generated for a containing object when one of its elements is removed. For example, when registering a Virtual Circuit (VC), IRemoveNotification is sent whenever a VC is removed. However, when registering on a specific VC, an IObjectDeletedNotification is sent only when the VC is removed. As another example is, if an NE is removed from a map, IRemoveNotification is generated, and if the map is removed, IObjectDeleteNotification is generated.

Unregistering from Notifications

After registering to one or more notifications, a client application should unregister when the notifications are no longer needed.

- To unregister by ID over a secured socket connection:

  `unregister` *ID*
- To unregister all over a secured socket connection:
  `Unregister all`
- To unregister over web services, use the void unregister (String registrationId) function. See Prime Network Web Services Endpoint References, page 74 for details.

Note Closing the socket connection implicitly unregisters all registrations.

5.2.3 Notification Interfaces Summary

Table 5-1 describes the notification operations supported by Prime Network.

Table 5-1 Supported Notification Operations

Operation	Description
register	Register for notification service.
register by ID	Register for notification service based on registration ID.
unregister by ID	Unregister notification service based on registration ID.
unregister by all	Unregister all notification service.

5.2.4 Sample BQL Scripts for Notification Service

This section contains the following sample BQL scripts:

- Registering for Notification Using the BQL Get and Register Command, page 104
-
- Registering for Notification Using the BQL Register **Command**, page 109

Registering for Notification Using the BQL Get and Register Command

In the following example, the client application is registering for notification using the command ID (111) for a particular map (map ID 58003). The client application will receive notification whenever this map is updated.

Here, **Get** and **Register** commands are used. When you run these commands, the information for the map ID 58003 is retrieved, and then the client application is registered to receive notification.

```
commandId=111
<?xml version="1.0" encoding="UTF-8"?>
<command name="Get">
  <param name="oid">
        <value>{[Map(Id=58003)]}</value>
  </param>
  <param name="rs">
        <value><key name="GetMapDetails">
  <entry name="depth">100</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">true</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.IHierarchyNode">
          <entry name="Name"/>
          <entry name="ManagedParent"/>
          <entry name="Map"/>
          <entry name="Children"/>
          <entry name="Leaf"/>
          <entry name="ContainedObjectOid"/>
        </key>
```

```
<key name="com.sheer.imo.IReconciliationAspect">
  <entry name="TypeEnum"/>
</key>
  <key name="com.sheer.imo.newalarm.ITicket">
        <entry name="Source"/>
        <entry name="EventCount"/>
        <entry name="LatestState"/>
        <entry name="AggregatedAckStateEnum"/>
        <entry name="AutoCleared"/>
        <entry name="Archived"/>
            <entry name="LastModificationTime"/>
            <entry name="AggregatedSeverityEnum"/>
            <entry name="ReductionCount"/>
            <entry name="DuplicationCount"/>
            <entry name="AffectedDevicesCount"/>
            <entry name="AlarmCount"/>
        </key>
        <key name="com.sheer.imo.ILink">
            <entry name="BiDirectional"/>
            <entry name="ConnectionInformation"/>
            <entry name="LinkTypeEnum"/>
        </key>
        <key name="com.sheer.imo.IVlanEntryReferencedStpAspect">
            <entry name="StpPortInfo"/>
        </key>
        <key name="com.sheer.imo.INE">
            <entry name="ScriptMetadataOids"/>
        </key>
        <key name="com.sheer.imo.IHierarchyChildrenAspect">
            <entry name="Children"/>
        </key>
        <key name="com.sheer.imo.IBusinessObject">
            <entry name="Name"/>
            <entry name="Notes"/>
            <entry name="TypeEnum"/>
            <entry name="EKey"/>
        </key>
        <key name="com.sheer.imo.IBridge">
            <entry name="StpInstanceInfo"/>
            <entry name="ScriptMetadataOids"/>
        </key>
        <key name="com.sheer.imo.IMapDataAspect">
            <entry name="LinksTypeFilter"/>
        </key>
        <key name="com.sheer.imo.IContainedImo">
            <entry name="ContainedObjectOid"/>
        </key>
        <key name="com.sheer.imo.IReferencedImosAspect">
            <entry name="Efd"/>
        </key>
        <key name="com.sheer.imo.IReferencedPortInfoAspect">
            <entry name="*"/>
        </key>
        <key name="com.sheer.imo.IReferencedEndPointsAspect">
            <entry name="ZEndPoint"/>
            <entry name="AEndPoint"/>
        </key>
```

```
<key name="com.sheer.imo.newalarm.ITicketListAspect">
    <entry name="Tickets"/>
</key>
<key name="com.sheer.imo.ITopologicalLink">
    <entry name="MaintenanceMode"/>
    <entry name="DetectionTypeEnum"/>
    <entry name="BiDirectional"/>
    <entry name="ConnectionInformation"/>
    <entry name="LinkTypeEnum"/>
</key>
<key name="com.sheer.imo.IStpPortsAspect">
    <entry name="*"/>
</key>
<key name="com.sheer.imo.IContainedHierarchyNode">
    <entry name="HierarchyRootOid"/>
</key>
<key name="com.sheer.imo.IMapAspect">
    <entry name="XCoordinate"/>
    <entry name="YCoordinate"/>
    <entry name="MapOid"/>
    <entry name="Height"/>
    <entry name="Background"/>
    <entry name="SubGraph"/>
    <entry name="Width"/>
</key>
<key name="com.sheer.imo.technologies.IIPInterface">
    <entry name="Address"/>
</key>
<key name="com.sheer.imo.IEthernetFlowDomain">
    <entry name="Name"/>
</key>
<key name="com.sheer.imo.IBusinessElement">
    <entry name="*"/>
</key>
<key name="com.sheer.imo.IReferencedBridgeAspect">
    <entry name="Bridge"/>
</key>
<key name="com.sheer.imo.IBusinessLink">
    <entry name="ZEndPoint"/>
    <entry name="AEndPoint"/>
    <entry name="BiDirectional"/>
    <entry name="LinkTypeEnum"/>
</key>
<key name="com.sheer.imo.IAlarmBusinessObject">
    <entry name="*"/>
</key>
<key name="com.sheer.imo.IRepAspect">
    <entry name="*"/>
</key>
<key name="com.sheer.imo.technologies.IStpPortInfo">
    <entry name="StpPortState"/>
    <entry name="StpPortRole"/>
    <entry name="ScriptMetadataOids"/>
</key>
<key name="com.sheer.imo.IReferencedStpInstanceInfoAspect">
    <entry name="StpInstanceInfo"/>
</key>
```

```
        <key name="com.sheer.imo.IVrf">
            <entry name="ScriptMetadataOids"/>
        </key>
        <key name="com.sheer.imo.technologies.IVlanEntry">
            <entry name="StpPortInfo"/>
            <entry name="ScriptMetadataOids"/>
        </key>
        <key name="com.sheer.imo.IReferencedStpPortsInfoAspect">
            <entry name="ASideStpPortInfo"/>
            <entry name="ZSideStpPortInfo"/>
        </key>
        <key name="com.sheer.imo.IMapLinksAspect">
            <entry name="Links"/>
        </key>
        <key name="com.sheer.imo.technologies.IStpInstanceInfo">
            <entry name="IsRoot"/>
            <entry name="ScriptMetadataOids"/>
        </key>
        <key name="com.sheer.imo.ISeverityAspect">
            <entry name="*"/>
        </key>
        <key name="com.sheer.imo.ICapacityExceededAspect">
            <entry name="*"/>
        </key>
        <key name="com.sheer.imo.IMap">
            <entry name="Name"/>
            <entry name="Links"/>
            <entry name="HierarchyRootOid"/>
        </key>
        <key name="com.sheer.imo.technologies.vendors.cisco.IREPPortInfo">
            <entry name="SegmentId"/>
            <entry name="PortRole"/>
            <entry name="BlockedVlans"/>
            <entry name="PortType"/>
            <entry name="OperPortStatus"/>
        </key>
        <key name="com.sheer.imo.IEthFlowPointReferencedNEAspect">
            <entry name="NetworkElement"/>
        </key>
        <key name="com.sheer.imo.IManagedElement">
            <entry name="SysLocation"/>
            <entry name="DeviceName"/>
            <entry name="ElementType"/>
            <entry name="IP"/>
            <entry name="SysUpTime"/>
            <entry name="VendorEnum"/>
            <entry name="ElementTypeKey"/>
            <entry name="ScriptMetadataOids"/>
            <entry name="CommunicationStateEnum"/>
            <entry name="ElementCategoryEnum"/>
            <entry name="SoftwareVersion"/>
            <entry name="SysDescription"/>
            <entry name="InvestigationStateEnum"/>
            <entry name="SysName"/>
        </key>
    </key>
    <key name="requiredAspects">
```

```
        <key name="com.sheer.imo.keys.IManagedElementOid">
            <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
            <entry name="com.sheer.imo.keys.ISeverityAspectOid"/>
        </key>
        <key name="com.sheer.imo.keys.IMapOid">
            <entry name="com.sheer.imo.keys.IContainedHierarchyNodeOid"/>
            <entry name="com.sheer.imo.keys.IMapDataAspectOid"/>
            <entry name="com.sheer.imo.keys.IMapLinksAspectOid"/>
            <entry name="com.sheer.imo.keys.ICapacityExceededAspectOid"/>
        </key>
        <key name="com.sheer.imo.keys.IBusinessElementOid">
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
        </key>
        <key name="com.sheer.imo.keys.IVlanEntryOid">
            <entry name="com.sheer.imo.keys.IReconciliationAspectOid"/>
            <entry
name="com.sheer.imo.keys.IVlanEntryReferencedStpAspectOid"/>
            <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
            <entry name="com.sheer.imo.keys.ISeverityAspectOid"/>
        </key>
        <key name="com.sheer.imo.keys.ILinkOid">
            <entry name="com.sheer.imo.keys.IRepAspectOid"/>
            <entry name="com.sheer.imo.keys.IStpPortsAspectOid"/>
            <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
            <entry name="com.sheer.imo.keys.ISeverityAspectOid"/>
        </key>
        <key name="com.sheer.imo.keys.IBridgeOid">
            <entry name="com.sheer.imo.keys.IReconciliationAspectOid"/>
            <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
            <entry
name="com.sheer.imo.keys.IReferencedStpInstanceInfoAspectOid"/>
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
            <entry name="com.sheer.imo.keys.ISeverityAspectOid"/>
        </key>
        <key name="com.sheer.imo.keys.INewAlarmOid">
            <entry name="com.sheer.imo.keys.IAlarmBusinessObjectOid"/>
        </key>
        <key name="com.sheer.imo.keys.IHierarchyNodeOid">
            <entry name="com.sheer.imo.keys.IMapAspectOid"/>
            <entry name="com.sheer.imo.keys.IHierarchyChildrenAspectOid"/>
            <entry name="com.sheer.imo.keys.IContainedImoOid"/>
        </key>
        <key name="com.sheer.imo.keys.IEthFlowPointOid">
            <entry
name="com.sheer.imo.keys.IEthFlowPointReferencedNEAspectOid"/>
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
        </key>
        <key name="com.sheer.imo.keys.ISwitchingEntityOid">
            <entry name="com.sheer.imo.keys.IReferencedBridgeAspectOid"/>
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
        </key>
        <key name="com.sheer.imo.keys.INetworkVlanOid">
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
            <entry name="com.sheer.imo.keys.IReferencedImosAspectOid"/>
```

```
        </key>
        <key name="com.sheer.imo.keys.IRepAspectOid">
            <entry
name="com.sheer.imo.keys.IReferencedPortInfoAspectOid"/>
        </key>
        <key name="com.sheer.imo.keys.IStpPortsAspectOid">
            <entry
name="com.sheer.imo.keys.IReferencedStpPortsInfoAspectOid"/>
        </key>
        <key name="com.sheer.imo.keys.IBusinessLinkOid">
            <entry
name="com.sheer.imo.keys.IReferencedEndPointsAspectOid"/>
        </key>
        <key name="com.sheer.imo.keys.INEOid">
            <entry name="com.sheer.imo.keys.IReconciliationAspectOid"/>
            <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
            <entry name="com.sheer.imo.keys.ISeverityAspectOid"/>
        </key>
    </key>
</key></value>
    </param>
</command>
```

Registering for Notification Using the BQL Register Command

In the following example, the client application is registering for notification of ticket updates for all network elements that are managed in Prime Network.

Here, the **Register** command is used. When you run this command, the client application is registered to receive notification without retrieving the data.

```
<command name="Register">
<param name="oid">
<value>{[TicketListAspect]}</value>
</param>
<param name="rs">
<value>
<key name="">
  <entry name="depth">100</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.newalarm.ITicketListAspect">
              <entry name="Tickets" />
        </key>
  <key name="com.sheer.imo.newalarm.ITicket">
        <entry name="*"/>
      </key>
  </key>
</key>
</value>
</param>
</command>
.
```

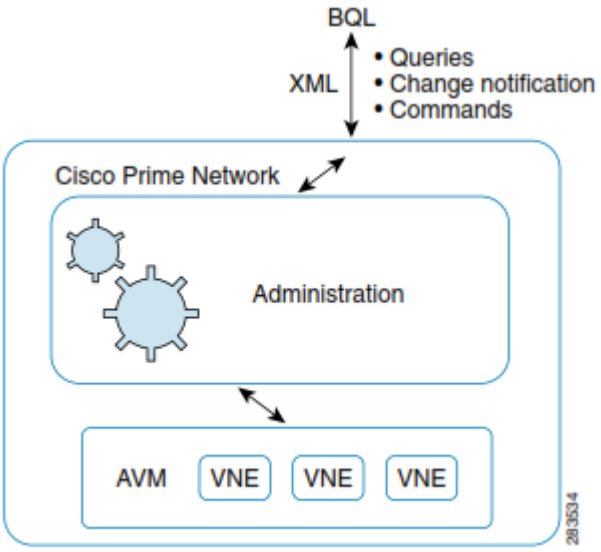
5.3 Administering Cisco Prime Network Using BQL

Administrators can control and configure the behavior of Cisco Prime Network using BQL commands. BQL interacts with the Prime Network Registry to query and modify configuration information. You can perform the following administration functions using BQL:

- Setting up and administrating Prime Network Servers (units and gateway).
- Managing global settings for viewing database segments, viewing and installing client licenses, customizing a message of the day, configuring polling groups, and configuring protection groups.
- Managing security, scopes, and user accounts.
- Customizing static links used in Prime Network Vision topology.

Figure 5-3 shows a high-level overview of how BQL is used to administer Prime Network.

Figure 5-3 Administering Prime Network Using BQL



AVM and VNEs

Prime Network gateway processes are divided into AVMs. AVMs are Java processes (independent JVMs) with their own dedicated memory. AVMs are mostly used to provide the necessary distribution support platform for executing and monitoring multiple VNEs. Prime Network gateway processes and VNEs commonly reside on Prime Network units, but they can also reside on a Prime Network gateway. Some Prime Network server processes are reserved for system use.

Virtual Network Elements (VNEs) are entities that maintain a live model of each network element and of the entire network. A VNE:

- Is an abstract representation of an actual network element that runs as a completely autonomous process within a Prime Network unit. Each VNE is assigned to manage a single network element instance and contains a replica of that element.
- Uses whatever southbound management interfaces the network element implements (for example, SNMP or Telnet).
    - Maintains a near-real-time network model using device polling and asynchronous data collection. The actual data collection, which is by nature vendor and device specific, is the responsibility of the VNE driver component of the VNE. A VNE driver maps specific device properties to a simplified vendor-neutral device model.

For more information about the Prime Network architecture, VNEs, and AVMs, see the *Cisco Prime Network 4.2.2 Administrator Guide*.

Additional Reading

- Review *Cisco Prime Network 4.2.2 Administrator Guide* to understand the Prime Network administrative tasks. This guide also provides details on the Prime Network user roles and scopes.
- See the *Cisco Prime Network Information Model* Javadoc to understand the IMO for the Prime Network administrative tasks. This document is available on the Prime Network Technology Center website. You must have a Cisco.com account with partner level access, or you must be a Prime Network licensee to access this website.

5.3.1 Management Interfaces

Table 5-2 lists the BQL commands supported for management-related queries.

Table 5-2 Supported Management BQL Command Queries

Command Name	IMO/OID Type	IMO/OID Value	Description
Get	com.sheer.imo.keys.IMCVMOid	{[MCNetwork][MCVM(IP=IPaddress)]}	Retrieves information on Prime Network units and gateway along with the list of AVMs created under it.
Get	com.sheer.imo.keys.IMCNetwork Oid	{[MCNetwork]}	Retrieves a list of Prime Network units and gateway.
Get	com.sheer.imo.keys.IAvmOid	{[MCNetwork][MCVM(IP=IPaddress)][Avm(AvmNumber=AVMNumber)]}	Retrieves information on a specific AVM along with the list of VNEs created under it.
Get	com.sheer.imo.keys.IPollingGroups ManagementOid	{[PollingGroupsManagement]}	Retrieves information on the list of available polling groups.
Get	com.sheer.imo.keys.IElement ManagementOid	{[MCNetwork][MCVM(IP=IPaddress)][])][Avm(AvmNumber=AVMNumber)][Element	Retrieves information on a specific VNE.

Command Name	IMO/OID Type	IMO/OID Value	Description
Get	com.sheer.imo.keys.IScopeRootOid	{[ScopeRoot]}	Retrieves a list of all available scopes.
Get	com.sheer.imo.keys.IScopeOid	{[Scope(Name=*ScopeName*)]}	Retrieves information on a specific scope.
Get	com.sheer.imo.management.ISchemes	{[Schemes]}	Retrieves a list of all supported schemes.
GetRole	—	—	Retrieves roles supported by Prime Network.

Table 5-3 lists the BQL commands supported for management-related operations.

Table 5-3 Supported Management BQL Commands Operations

Command Name	IMO/OID Type	IMO/OID Value	Description
Create	com.sheer.imo.management.IMC	{[MCNetwork][MCVM(IP=<ipaddress>)]}	Creates a new Prime Network unit.
Update	com.sheer.imo.keys.IMCVMOid	{[MCNetwork][MCVM(IP=<ipaddress>)]}	Updates an existing Prime Network unit.
Delete	com.sheer.imo.keys.IMCVMOid	{[MCNetwork][MCVM(IP=<ipaddress>)]}	Removes an existing Prime Network unit.
MCRestart	com.sheer.imo.keys.IMCVMOid	{[MCNetwork][MCVM(IP=<ipaddress>)]}	Restarts an existing Prime Network unit.
Create	com.sheer.imo.management.IAvm	{[MCNetwork][MCVM(IP=<ipaddress>)][Avm(AvmNumber=<avm_no>)]}	Creates a new AVM.
Load	com.sheer.imo.keys.IAvmOid	{[MCNetwork][MCVM(IP=<ipaddress>)][Avm(AvmNumber=<avm_no>)]}	Starts an existing AVM process.
Unload	com.sheer.imo.keys.IAvmOid	{[MCNetwork][MCVM(IP=<ipaddress>)][Avm(AvmNumber=<avm_no>)]}	Stops an existing AVM process.
Update	com.sheer.imo.management.IAvmOid	{[MCNetwork][MCVM(IP=<ipaddress>)][Avm(AvmNumber=<avm_no>)]}	Updates an existing AVM.

Command Name	IMO/OID Type	IMO/OID Value	Description
Delete	com.sheer.imo.management.IAvmOid	{[MCNetwork][MCVM(IP=<ipaddress>)][Avm(AvmNumber=<avm_no>)]}	Removes an existing AVM.
Create	com.sheer.imo.keys.IElementManagementOid	{[MCNetwork][MCVM(IP=<ipaddress>)][])][Avm(AvmNumber=<avm_no>)][ElementManagement(Key=<vne_name>)]}	Creates a new VNE.
Update	com.sheer.imo.keys.IElementManagementOid	{[MCNetwork][MCVM(IP=<ipaddress>)][])][Avm(AvmNumber=<avm_no>)][ElementManagement(Key=<vne_name>)]}	Updates an existing VNE.
Delete	com.sheer.imo.keys.IElementManagementOid	{[MCNetwork][MCVM(IP=<ipaddress>)][])][Avm(AvmNumber=<avm_no>)][ElementManagement(Key=<vne_name>)]}	Removes an existing VNE.
Create	com.sheer.imo.IScope	{[Scope(Name=<scope_name>)]}	Creates a new scope.
Update	com.sheer.imo.keys.IScopeOid	{[Scope(Name=<scope_name>)]}	Updates an existing scope.
Delete	com.sheer.imo.keys.IScopeOid	{[Scope(Name=<scope_name>)]}	Deletes an existing scope.

Required OID Strings

Table 5-4 list the OID strings that are required to manage AVMs and VNEs.

Table 5-4 OID Strings Required for Managing AVMs and VNEs

Operation	Required OID String	Example
Add AVM	Prime Network server (unit)	{[MCNetwork][MCVM(IP=1.1.1.1)]}
Stop/start/delete AVM	AVM	{[MCNetwork][MCVM(IP=1.1.1.1)][Avm(AvmNumber=111)]}

Operation	Required OID String	Example
Add VNE	AVM Optional: Polling group	{[MCNetwork][MCVM(IP=1.1.1.1)][Avm(AvmNumber=111)]} {[PollingGroupsManagement][PollingGroupManagement(Name=default)]} Or {[MCNetwork][MCVM(IP=10.56.58.178)][Avm(AvmNumber=111)][ElementManagement(Key=vneName)][PollingGroupManagement(Name=default)]}
Stop/start/delete VNE	VNE	{[MCNetwork][MCVM(IP=1.1.1.1)][Avm(AvmNumber=111)][ElementManagement(Ke y=vneName)]}
Update VNE	VNE Optional: Polling group	{[MCNetwork][MCVM(IP=1.1.1.1)][Avm(AvmNumber=111)][ElementManagement(Ke y=vneName)]} {[PollingGroupsManagement][PollingGroupManagement(Name=slow)]} Or {[MCNetwork][MCVM(IP=10.56.58.178)][Avm(AvmNumber=111)][ElementManagement(Key=vneName)][PollingGroupManagement(Name=slow)]}

5.3.2 Sample BQL Scripts for Managing AVMs and VNEs

This section contains the following sample BQL scripts:

The Mediator Debugger tool helps you identify the BQL commands for any Prime Network GUI task (except for Prime Network Events tasks). Using this tool, you can write your own BQL commands for the required GUI tasks.

Creating a Prime Network Unit

The following example shows the usage of the BQL **Create** command to add a unit in the Prime Network gateway. Here, change the unit IP address (10.77.213.238) to add a new unit under the default protection group with the high availability feature enabled.

```
<command name="Create">
  <param name="imobject">
  <value>
  <management.IMC>
    <ID type="Oid">
      {[MCNetwork][MCVM(IP=10.77.213.238)]}
    </ID>
    <HighAvailabilityEnabled type="Boolean">true</HighAvailabilityEnabled>
    <ProtectionGroupOid type="Oid">
    {[MCNetwork][MCVM(IP=10.77.213.238)][ProtectionGroup(Key=default-pg)]}
    </ProtectionGroupOid>
  </management.IMC>
  </value>
  </param>
</command>
.
```

Creating an AVM

The following example shows the usage of the BQL **Create** command to add an AVM in the Prime Network gateway. Here, change the Prime Network unit IP address (10.77.213.238), AVM number (601), and AVM name (TestAVM) to add a new AVM with the high availability feature enabled.

```
<command name="Create">
  <param name="imobject">
  <value>
  <management.IAvm>
    <ID
  type="Oid">{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601)]}</I
  D>
    <AdminStatusEnum type="Integer">1</AdminStatusEnum>
```

```
    <AvmKey type="com.sheer.types.Key">TestAVM</AvmKey>
    <HighAvailabilityEnabled type="Boolean">true</HighAvailabilityEnabled>
    <MaximumMemory type="Integer">256</MaximumMemory>
  </management.IAvm>
  </value>
  </param>
</command>
```

Creating a VNE

The following example shows the usage of the BQL **Create** command to add a VNE. Here, change the Prime Network unit IP address (10.77.213.238), AVM number (601), VNE name (TestVNE), and VNE IP address (192.168.2.2) to add a new VNE. In this example:

- ICMP is enabled with polling rate (2000000 msec).
- Local Setting adaptive polling is enabled with upper threshold (90) and lower threshold (60) values.
- SNMPv1 is enabled along with the community strings (public and private).
- VNE scheme (product) is defined.
- Telnet is enabled with the telnet sequence (#,cisco,:,).
- Event generating IP address (192.168.10.10) is configured.

```
<command name="Create">
  <param name="imobject">
        <value>
              <management.IElementManagement>
        <ID type="Oid">{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumbe
  r=601)][ElementManagement(Key=TestVNE)]}</ID>
        <AdaptivePollingSettingEnum type="Integer">2</AdaptivePollingSet
tingEnum>
              <AdminStatusEnum type="Integer">0</AdminStatusEnum>
              <ElementClassEnum type="Integer">0</ElementClassEnum>
              <ElementName type="String">TestVNE</ElementName>
              <ICMPEnabled type="Boolean">true</ICMPEnabled>
              <ICMPPollingRate type="Integer">2000000</ICMPPollingRate>
              <IP type="com.sheer.types.IPAddress">192.168.2.2</IP>
<LocalAdaptivePollingSettings type="management.IAdaptivePollingS
ettings">
              <ID type="Oid">{[]}</ID>
              <Enabled type="Boolean">true</Enabled>
                    <LowerThreshold type="Integer">60</LowerThreshold>
                    <UpperThreshold type="Integer">90</UpperThreshold>
                 </LocalAdaptivePollingSettings>
                <MaintenanceEnabled
              type="Boolean">false</MaintenanceEnabled>
                <PollingGroup type="management.IPollingGroupManagement">
              <ID
        type="Oid">{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmN
        umber=601)][ElementManagement(Key=TestVNE)][PollingGroupManagemen
        t(Name=default)           ]}</ID>
              </PollingGroup>
              <SNMPEnabled type="Boolean">true</SNMPEnabled>
              <SNMPReadCommunity type="String">public</SNMPReadCommunity>
              <SNMPWriteCommunity
         type="String">private</SNMPWriteCommunity>
              <SchemeName type="String">product</SchemeName>
```

```
        <SnmpV3AuthenticationEnum
        type="Integer">0</SnmpV3AuthenticationEnum>
                  <SnmpV3AuthenticationPassword type="String" />
                  <SnmpV3AuthenticationUserProfile type="String" />
                  <SnmpV3EncryptionPassword type="String" />
                  <SnmpVersionEnum type="Integer">0</SnmpVersionEnum>
                  <TelnetEnabled type="Boolean">true</TelnetEnabled>
                  <TelnetPortNumber type="Integer">23</TelnetPortNumber>
                  <TelnetProtocolEnum
                type="Integer">0</TelnetProtocolEnum>
                  <TelnetSequence
                type="String">#,cisco,:,</TelnetSequence>
                  <TrapSyslogSources type="management.ITrapSyslogSources">
              <ID
        type="Oid">{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601
        )][ElementManagement(Key=TestVNE)][TrapSyslogSources]}</ID>
                     <IPAddresses type="com.sheer.types.IPAddress_Array">
                     <com.sheer.types.IPAddress>192.168.10.10</com.sheer.t
              ypes.IPAddress>
                     </IPAddresses>
                  </TrapSyslogSources>
              </management.IElementManagement>
            </value>
        </param>
</command>
.
```

Getting Prime Network Unit Details

The following example shows the usage of the BQL **Get** command to retrieve the details of Prime Network unit (10.77.213.238); that is, to retrieve details of all AVMs running in the Prime Network unit.

```
<command name="Get">
  <param name="oid">
  <value>
        {[MCNetwork][MCVM(IP=10.77.213.238)]}
  </value>
  </param>
  <param name="rs">
  <value>
        <key name="com.sheer.imo.keys.IMCVMOid">
        <entry name="depth">1</entry>
        <entry name="register">true</entry>
        <entry name="cachedResultAcceptable">false</entry>
        <key name="requiredProperties">
        <key name="com.sheer.imo.management.IMC">
        <entry name="*"/>
        </key>
        <key name="com.sheer.imo.management.IAvm">
        <entry name="*"/>
        </key>
        </key>
        <key name="excludedProperties">
        <key name="com.sheer.imo.management.IAvm">
        <entry name="ElementManagements"/>
        </key>
        </key>
        </key>
```

```
  </value>
  </param>
</command>
```

Getting AVM Details

The following example shows the usage of the BQL **Get** command to retrieve the details of AVM (601), managed in Prime Network unit (10.77.213.238); that is, to retrieve details of all VNEs running in the AVM.

```
<command name="Get">
  <param name="oid">
    <value>{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601)]}</val
  ue>
  </param>
  <param name="rs">
  <value><key name="com.sheer.imo.keys.IAvmOid">
  <entry name="depth">1</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.management.IElementManagement">
              <entry name="OperStatusEnum"/>
              <entry name="SNMPEnabled"/>
              <entry name="ElementType"/>
              <entry name="DeviceTypeAdaptivePollingSettings"/>
              <entry name="TrapSyslogSources"/>
              <entry name="IP"/>
              <entry name="AdaptivePollingSettingEnum"/>
              <entry name="TelnetEnabled"/>
              <entry name="AdminStatusEnum"/>
              <entry name="LocalAdaptivePollingSettings"/>
              <entry name="MaintenanceEnabled"/>
              <entry name="PollingGroup"/>
              <entry name="StartTime"/>
              <entry name="ElementClassEnum"/>
  </key>
  <key name="com.sheer.imo.management.IAvm">
        <entry name="*"/>
  </key>
</key>
<key name="excludedProperties">
  <key name="com.sheer.imo.management.IElementManagement">
        <entry name="LeadingMangementComponent"/>
  </key>
  <key name="com.sheer.imo.management.IPollingInterval">
        <entry name="*"/>
  </key>
  <key name="com.sheer.imo.management.IPowerDrillData">
        <entry name="*"/>
  </key>
  <key name="com.sheer.imo.management.IPollingGroupManagement">
        <entry name="*"/>
  </key>
</key>
</value>
  </param>
</command>
```

.

Getting AVM Properties

The following example shows the usage of the BQL **Get** command to retrieve the properties of AVM (601), managed in Prime Network unit (10.77.213.238).

```
<command name="Get">
  <param name="oid">
        <value>
              {[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601)]}
        </value>
  </param>
  <param name="rs">
  <value>
        <key name="IAVMOid-NoAgents">
        <entry name="depth">1</entry>
        <entry name="register">false</entry>
        <entry name="cachedResultAcceptable">false</entry>
        <key name="requiredProperties">
        <key name="com.sheer.imo.management.IAvm">
        <entry name="*"/>
        </key>
        </key>
        <key name="excludedProperties">
        <key name="com.sheer.imo.management.IAvm">
        <entry name="ElementManagements"/>
        </key>
        </key>
        </key>
  </value>
  </param>
</command>
```

Getting AVM Memory Details

The following example shows the usage of the BQL **GetMemoryUsage** command to retrieve the memory usage of AVM (400), managed in Prime Network unit (10.56.22.93).

```
<command name="GetMemoryUsage">
  <param name="oid">
        <value>
              {[MCNetwork][MCVM(IP=10.56.22.93)][Avm(AvmNumber=400)]}
        </value>
  </param>
</command>
```

.

Getting VNE Properties

The following example shows the usage of the BQL **Get** command to retrieve the properties of VNE (10.77.214.134), belonging to AVM (601), managed in Prime Network unit (10.77.213.238).

```
<command name="Get">
  <param name="oid">
```

```
<value>{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601)][ElementMan
agement(Key=10.7
7.214.134)]}</value>
  </param>
  <param name="rs">
        <value><key name="com.sheer.imo.keys.IElementManagementOid">
  <entry name="depth">1</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.management.IElementManagement">
              <entry name="*"/>
        </key>
        <key name="com.sheer.imo.management.IPollingInterval">
              <entry name="*"/>
        </key>
        <key name="com.sheer.imo.management.IPowerDrillData">
              <entry name="*"/>
        </key>
        <key name="com.sheer.imo.management.ITrapSyslogSources">
              <entry name="*"/>
        </key>
        <key name="com.sheer.imo.management.IAdaptivePollingSettings">
              <entry name="*"/>
        </key>
        <key name="com.sheer.imo.management.IPollingGroupManagement">
              <entry name="*"/>
        </key>
      </key>
      <key name="excludedProperties">
        <key name="com.sheer.imo.management.IElementManagement">
              <entry name="LeadingMangementComponent"/>
        </key>
      </key>
</key></value>
  </param>
</command>
.
```

Getting Prime Network Unit Properties

The following example shows the usage of the BQL **Get** command to retrieve the properties of Prime Network unit (10.77.213.238).

```
<command name="Get">
  <param name="oid">
        <value>{[MCNetwork][MCVM(IP=10.77.213.238)]}</value>
  </param>
  <param name="rs">
        <value><key name="IMCVMOid-NoAVMs">
  <entry name="depth">1</entry>
  <entry name="register">false</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.management.IMC">
              <entry name="*"/>
        </key>
  </key>
```

```
  <key name="excludedProperties">
        <key name="com.sheer.imo.management.IMC">
              <entry name="Avms"/>
        </key>
  </key>
</key></value>
  </param>
</command>
.
```

Getting Prime Network Gateway Properties

The following example shows the usage of the BQL **Get** command to retrieve the properties of Prime Network gateway.

```
<command name="Get">
  <param name="oid">
  <value>
        {[MCNetwork]}
  </value>
  </param>
  <param name="rs">
  <value>
  <key name="com.sheer.imo.keys.IMCNetworkOid">
  <entry name="depth">1</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
  <key name="com.sheer.imo.management.IMC">
  <entry name="*"/>
  </key>
  <key name="com.sheer.imo.management.IMCNetwork">
  <entry name="MetroCentrals"/>
  </key>
  </key>
  <key name="excludedProperties">
  <key name="com.sheer.imo.management.IMC">
  <entry name="Avms"/>
  </key>
  </key>
  </key>
  </value>
  </param>
</command>
.
```

Getting Polling Group Details

The following example shows the usage of the BQL **Get** command to retrieve the polling group details that you have defined.

```
<command name="Get">
  <param name="oid">
  <value>
        {[PollingGroupsManagement]}
  </value>
  </param>
  <param name="rs">
```

```
  <value>
        <key name="com.sheer.imo.keys.IPollingGroupsManagementOid">
        <entry name="depth">100</entry>
        <entry name="register">true</entry>
        <entry name="cachedResultAcceptable">false</entry>
        <key name="requiredProperties">
        <key name="com.sheer.imo.IMO">
        <entry name="*"/>
        </key>
        </key>
        <key name="requiredAspects">
        <key name="com.sheer.imo.keys.IOid">
        <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
        </key>
        </key>
        </key>
  </value>
  </param>
</command>
.
```

Getting a Role

The following example shows the usage of the BQL **GetRole** command to retrieve the command builder script roles (administrator, viewer, and so on).

```
<command name="GetRole" />
.
```

Starting an AVM

The following example shows the usage of the BQL **Load** command to start an AVM (601), managed in the Prime Network unit (10.77.213.238).

```
<command name="Load">
  <param name="oid">
  <value>
        {[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601)]}
  </value>
  </param>
</command>
.
```

Stopping an AVM

The following example shows the usage of the BQL **Unload** command to stop an AVM (601), managed in the Prime Network unit (10.77.213.238).

```
<command name="Unload">
  <param name="oid">
        <value>{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=610)]}</
      value>
  </param>
</command>
.
```

Starting a VNE

The following example shows the usage of the BQL **Update** command to start a VNE (TestVNE), belonging to AVM (601), managed in Prime Network unit (10.77.213.238). Here, the VNE status is defined as part of IScalarNotification (AdminStatusEnum type value 1 [Up] and 0 [Down]). See IScalarNotification, page 93 for more details.

```
<command name="Update">
  <param name="oid">
  <value>{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601)][ElementM
anagement(Key=Test VNE)]}</value>
  </param>
  <param name="imobjectArr">
        <value>
              <IScalarNotification>
                    <NewIMO type="management.IElementManagement">
                        <ID
                   type="Oid">{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmN
                   umber=601)][ElementManagement(Key= TestVNE)]}</ID>
                         <AdminStatusEnum
                    type="Integer">1</AdminStatusEnum>
                    </NewIMO>
                    <OldIMO type="management.IElementManagement">
                          <ID
                    type="Oid">{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(Av
                    mNumber=601)][ElementManagement(Key= TestVNE)]}</ID>
                          <AdminStatusEnum
                       type="Integer">0</AdminStatusEnum>
                     </OldIMO>
                     <PropertyName
                type="String">AdminStatusEnum</PropertyName>
              </IScalarNotification>
        </value>
  </param>
</command>
.
```

Stopping a VNE

The following example shows the usage of the BQL **Update** command to start a VNE (TestVNE), belonging to AVM (601), managed in Prime Network unit (10.77.213.238). Here, the VNE status is defined as part of IScalarNotification (AdminStatusEnum type value 0 [Down] and 1 [Up]). See IScalarNotification, page 93 for more details.

```
<command name="Update">
  <param name="oid">
    <value>{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601)][Elemen
  tManagement( Key=TestVNE)]}</value>
  </param>
  <param name="imobjectArr">
  <value>
  <IScalarNotification>
        <NewIMO type="management.IElementManagement">
        <ID
      type="Oid">{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601)][E
      lementManagem ent(Key=TestVNE)]}</ID>
        <AdminStatusEnum type="Integer">0</AdminStatusEnum>
        </NewIMO>
        <OldIMO type="management.IElementManagement">
```

```
        <ID
    type="Oid">{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601)][El
    ementManagem ent(Key=TestVNE)]}</ID>
        <AdminStatusEnum type="Integer">1</AdminStatusEnum>
        </OldIMO>
        <PropertyName type="String">AdminStatusEnum</PropertyName>
      </IScalarNotification>
  </value>
</param>
</command>
.
```

Disabling Adaptive Polling

The following example shows the usage of the BQL **Update** command to disable the adaptive polling for an NE 169.254.198.143. In this example, the adaptive polling is disabled and local setting is enabled with Upper Threshold and Lower Threshold value as 90 and 60 respectively.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Update">
  <param name="oid">
<value>{[MCNetwork][MCVM(IP=10.56.56.117)][Avm(AvmNumber=555)][ElementMana
gement(Key=169.254.198.143)]}</value>
  </param>
  <param name="imobjectArr">
        <value>
              <IScalarNotification>
                    <NewIMO type="management.IElementManagement">
                          <ID
                          type="Oid">{[MCNetwork][MCVM(IP=10.56.56.117)][A
                          vm(AvmNumber=555)][ElementManagement(Key=1
                          69.254.198.143)]}</ID>
                    <AdaptivePollingSettingEnum
              type="Integer">2</AdaptivePollingSettingEnum>
                    </NewIMO>
                    <OldIMO type="management.IElementManagement">
                          <IDtype="Oid">{[MCNetwork][MCVM(IP=10.56.56.117)
                        ][Avm(AvmNumber=555)][ElementManagement(Key=169.25
                        4.198.143)]}</ID>
                    <AdaptivePollingSettingEnum
                    type="Integer">1</AdaptivePollingSettingEnum>
                    </OldIMO>
                    <PropertyName
              type="String">AdaptivePollingSettingEnum</PropertyName>
              </IScalarNotification>
        </value>
      <value>
        <IScalarNotification>
              <NewIMO type="management.IElementManagement">
                    <IDtype="Oid">{[MCNetwork][MCVM(IP=10.56.56.117)][Avm(
                AvmNumber=555)][ElementManagement(Key=169.254.198.143)]}</
                ID>
                      <LocalAdaptivePollingSettings
                type="management.IAdaptivePollingSettings">
                          <ID type="Oid">{[]}</ID>
                          <Enabled type="Boolean">false</Enabled>
```

```
                        <LowerThreshold
                        type="Integer">60</LowerThreshold>
                        <UpperThreshold
                        type="Integer">90</UpperThreshold>
                    </LocalAdaptivePollingSettings>
              </NewIMO>
              <OldIMO type="management.IElementManagement">
                    <IDtype="Oid">{[MCNetwork][MCVM(IP=10.56.56.117)][Avm(
                  AvmNumber=555)][ElementManagement(Key=169.254.198.143)]}
                  </ID>
                    <LocalAdaptivePollingSettings
              type="">Null</LocalAdaptivePollingSettings>
              </OldIMO>
              <PropertyName
        type="String">LocalAdaptivePollingSettings</PropertyName>
        </IScalarNotification>
      </value>
  </param>
</command>
.
```

Deleting a VNE

The following example shows the usage of the BQL **Delete** command to delete a VNE (TestVNE), belonging to AVM (601), managed in Prime Network unit (10.77.213.238).

```
<command name="Delete">
  <param name="oids">
<value>{[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601)][ElementMan
agement(Key=TestVNE)]}</value>
  </param>
  <param name="signature">
        <value>true</value>
  </param>
</command>
.
```

Deleting an AVM

The following example shows the usage of the BQL **Delete** command to delete an AVM (601), managed in Prime Network unit (10.77.213.238).

```
<command name="Delete">
  <param name="oid">
  <value>
        {[MCNetwork][MCVM(IP=10.77.213.238)][Avm(AvmNumber=601)]}
  </value>
  </param>
</command>
.
```

Deleting a Prime Network Unit

The following example shows the usage of the BQL **Delete** command to delete the Prime Network unit (10.77.213.238).

```
<command name="Delete">
  <param name="oids">
```

```
  <value>
        {[MCNetwork][MCVM(IP=10.77.213.238)]}
  </value>
  </param>
</command>
.
```

5.4 Retrieving Inventory Data Using BQL

Physical and logical inventory data can be retrieved using BQL commands. BQL commands allow you to:

- Access network elements; for example, you can get all network elements or the network topology and connectivity information.
- Access physical inventory information; for example, you can get the complete physical containment for a network element or the software assets running on a network element or on a specified module.
- Access logical inventory information; for example, you can get the network layer protocol information or all physical termination points associated to network elements.

Figure 5-4 shows a high-level overview of how BQL can be used to retrieve inventory data.

Figure 5-4 Retrieving Inventory Data Using BQL

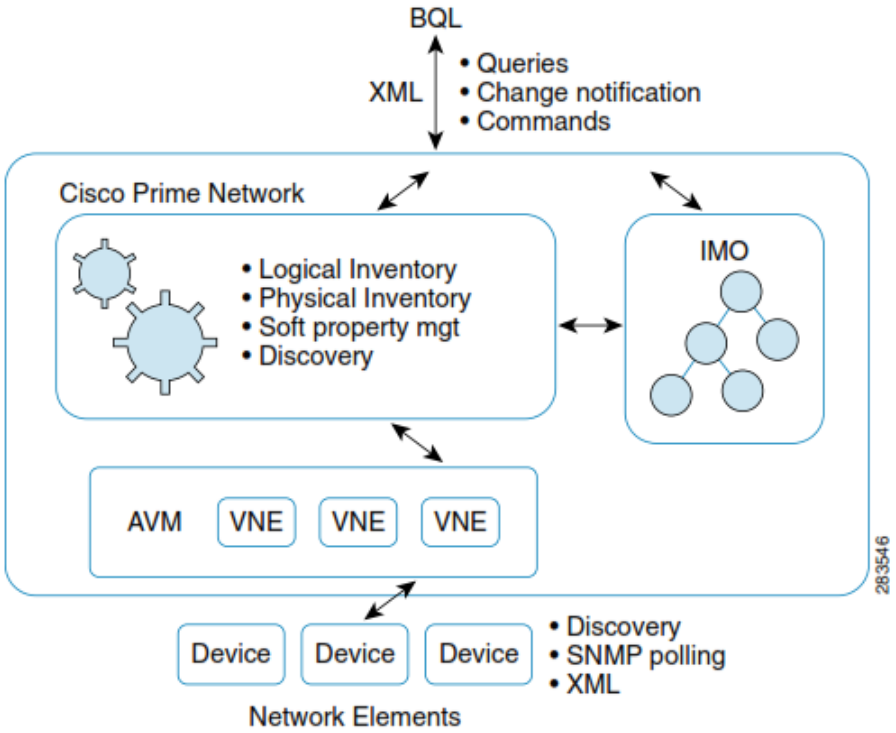
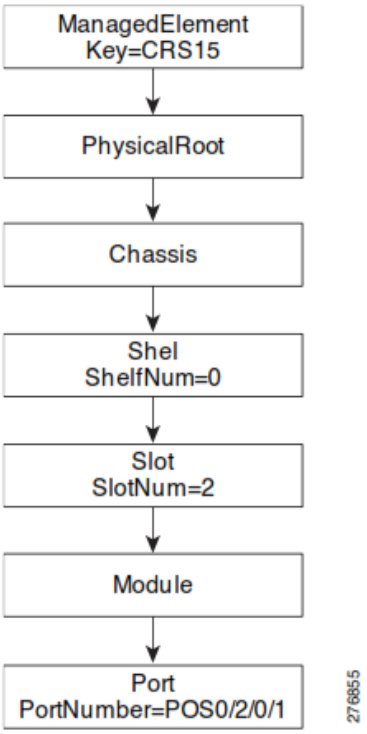


Figure 5-5 and Figure 5-6 illustrate the physical and logical object hierarchy. The physical object of a Cisco CRS1 port is:

```
{[ManagedElement(Key=CRS15)][PhysicalRoot][Chassis][Shelf(ShelfNum=0)][Slot
(SlotNum=2)][Mo dule][Port(PortNumber=POS0/2/0/1)][PhysicalLayer]}
```

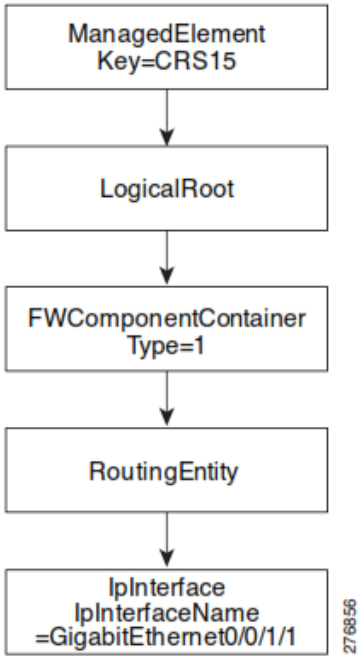
Figure 5-5 Physical Object Hierarchy



The logical object of a Cisco CRS1 is:

```
{[ManagedElement(Key=CRS15)][LogicalRoot][FWComponentContainer(Type=1)][Ro
utingEntity][IpInterface(IpInterfaceName=GigabitEthernet0/0/1/1)]}
```

Figure 5-6 Logical Object Hierarchy



Additional Reading

- Review *Cisco Prime Network 4.2.2 Administrator Guide* to understand the Cisco Prime Network user roles and scopes.
- Review *Cisco Prime Network 4.2.2 User Guide* to understand the Prime Network applications, Cisco Prime Network Vision, and Cisco Prime Network Events.
- Review *Cisco Prime Network 4.2.2 Supported Cisco VNEs* and *Cisco Prime Network 4.2.2 Supported Technologies and Topologies* to learn about supported technologies, supported NEs, and topology links for the supported NEs.
- See the *Cisco Prime Network Information Model* Javadoc to understand the IMO for VNE, path tracer, business tags, NEs, and maps. This document is available on the Prime Network Technology Center website. You must have a Cisco.com account with partner level access, or you must be a Prime Network licensee to access this website.

5.4.1 Inventory Interfaces

Table 5-5 lists the BQL commands supported for inventory.

Table 5-5 Supported Inventory BQL Command Queries

Command Name	IMO/OID Type	IMO/OID Value	Description
Get	com.sheer.imo.IManagedElement	{[ManagedElement(Key=*Device Name*)]}	Retrieves complete physical and logical inventory details for the requested network element.
DeviceList	—	—	Retrieves all managed network elements in Prime Network.
Physical Inventory			
Get	com.sheer.imo.IPhysicalRoot	{[ManagedElement(Key=*Device Name*)][PhysicalRoot]}	Retrieves complete physical inventory details for the requested network element.
Get	com.sheer.imo.IChassis	{[ManagedElement (Key=*DeviceName*)] [PhysicalRoot][Chassis]}	Retrieves the chassis details for the requested network element.
Get	com.sheer.imo.IEquipmentHolder	{[ManagedElement(Key=*Device Name*)][PhysicalRoot][Chassis] [Slot(SlotNum=*SlotNumber*)]}	Retrieves the slot details for the requested network element.
Get	com.sheer.imo.IModule	{[ManagedElement(Key=*Device Name*)][PhysicalRoot][Chassis] [Slot(SlotNum=*SlotNumber*)] [Module]}	Retrieves the module details for the requested network element.
Get	com.sheer.imo.IPortConnector	{[ManagedElement(Key=*Device Name*)][PhysicalRoot][Chassis] [Slot(SlotNum=*SlotNumber*)] [Module][Slot(SlotNum=*Slot Number*)][Module][Port	Retrieves the port details for the requested network element.
Get	com.sheer.imo.IShelf	{[ManagedElement(Key=*Device Name*)][PhysicalRoot][Chassis] [Shelf(ShelfNum=*ShelfNumber*)]}	Retrieves the shelf details for the requested network element.
Logical Inventory			

Command Name	IMO/OID Type	IMO/OID Value	Description
Get	com.sheer.imo.ILogicalRoot	{[ManagedElement(Key=*Device Name*)][LogicalRoot]}	Retrieves complete logical inventory details for the requested network element.

See Processing BQL Notification Messages, page 87 to understand how to register for notification messages whenever there is an update in the inventory data.
See Managing Soft Properties Using BQL, page 196 to understand how to extend the NE data collection and modeling by adding new properties to the DCs, and assigning them to NE MIB variables in runtime.

5.4.2 Sample BQL Scripts for Retrieving Inventory Data

This section contains the following sample BQL scripts:

- Retrieving an NE List, page 132
- Retrieving NE Properties, page 133
- Retrieving Physical Inventory Data for an NE, page 133
- Retrieving Physical Inventory Without Ports, page 133
- Retrieving Physical Inventory with Ethernet Ports, page 134
- Retrieving NE Port Status, page 135
- Retrieving ARP Entries, page 135
- Retrieving All Pseudowires, page 136
- Retrieving All VSIs, page 137
- Retrieving a Specific VSI, page 137
- Refreshing Inventory Data for an NE, page 138
- Finding the Path Trace, page 138
- Attaching **a Business Tag**, page 140
- Editing a Business Tag, page 140
- Finding All Maps, page 141
- Getting Map Details, page 141

The Mediator Debugger tool helps you identify the BQL commands for any Prime Network GUI task (except for Cisco Prime Network Events tasks). Using this tool, you can write your own BQL commands for the required GUI tasks.

Retrieving an NE List

The following example shows the usage of the BQL **DeviceList** command to retrieve all managed network elements in Prime Network.

```
<command name="DeviceList">
</command>
.
```

Retrieving NE Properties

The following example shows the usage of the BQL **Get** command to retrieve the network element (core-crs1-p2) properties.

```
<command name="Get">
  <param name="oid">
        <value>{[ManagedElement(Key=core-crs1-p2)]}</value>
  </param>
  <param name="rs">
        <value>
              <key name="NE report">
              <entry name="register">false</entry>
              <key name="requiredProperties">
              <key name="com.sheer.imo.IManagedElement">
              <entry name="*"/>
              </key>
              </key>
              </key>
        </value>
  </param>
</command>
.
```

Retrieving Physical Inventory Data for an NE

The following example shows the usage of the BQL **Get** command to retrieve the network element (core-crs1-p2) physical inventory data.

```
<command name="Get">
  <param name="oid">
  <value>
        {[ManagedElement(Key=core-crs1-p2)][PhysicalRoot]}
  </value>
  </param>
  <param name="rs">
  <value>
        <key name="NE report">
        <entry name="register">false</entry>
        <key name="requiredProperties">
        <key name="com.sheer.imo.IPhysicalRoot">
        <entry name="*"/>
        </key>
        <key name="com.sheer.imo.IEquipment">
        <entry name="*"/>
        </key>
        <key name="com.sheer.imo.IEquipmentHolder">
        <entry name="*"/>
        </key>
        </key>
        </key>
  </value>
  </param>
</command>
.
```

Retrieving Physical Inventory Without Ports

The following example shows the usage of the BQL **Get** command to retrieve the network element (core-crs1-p2) physical inventory data without the ports details (SupportedPTPs are excluded).

```
<command name="Get">
  <param name="oid">
  <value>
        {[ManagedElement(Key=core-crs1-p2)][PhysicalRoot]}
  </value>
  </param>
  <param name="rs">
  <value>
      <key name="NE report">
      <entry name="register">false</entry>
      <key name="requiredProperties">
      <key name="com.sheer.imo.IPhysicalRoot">
      <entry name="*"/>
      </key>
      <key name="com.sheer.imo.IEquipment">
      <entry name="*"/>
      </key>
      <key name="com.sheer.imo.IEquipmentHolder">
      <entry name="*"/>
      </key>
      </key>
      <key name="excludedProperties">
      <key name="com.sheer.imo.IModule">
      <entry name="SupportedPTPs"/>
      </key>
      </key>
      </key>
  </value>
  </param>
</command>
```

.

Retrieving Physical Inventory with Ethernet Ports

The following example shows the usage of the BQL **Get** command to retrieve the network element (core-crs1-p2) physical inventory data with Ethernet ports.

```
<command name="Get">
  <param name="oid">
  <value>
        {[ManagedElement(Key=core-crs1-p2)][PhysicalRoot]}
  </value>
  </param>
  <param name="rs">
  <value>
        <key name="NE report">
        <entry name="register">false</entry>
        <key name="requiredProperties">
        <key name="com.sheer.imo.IPhysicalRoot">
        <entry name="*"/>
        </key>
        <key name="com.sheer.imo.IEquipment">
        <entry name="*"/>
        </key>
```

```
        <key name="com.sheer.imo.IEquipmentHolder">
        <entry name="*"/>
        </key>
        <key name="com.sheer.imo.IPortConnector">
        <entry name="PortAlias"/>
        <entry name="ContainedCurrentCTPs"/>
        </key>
        <key name="com.sheer.imo.IPhysicalLayer">
        <entry name="TypeEnum"/>
        <entry name="ContainedCurrentCTPs"/>
        </key>
        <key name="com.sheer.imo.technologies.IEthernet">
        <entry name="*"/>
        </key>
        </key>
        </key>
  </value>
  </param>
</command>
.
```

Retrieving NE Port Status

The following example shows the usage of the BQL **Get** command to retrieve the port status (admin and oper) of a network element (core-crs1-p2).

```
<command name="Get">
  <param name="oid">
  <value>
    {[ManagedElement(Key=core-crs1-
  p2)][PhysicalRoot][Chassis][Shelf(ShelfNum=0)][Slot
  (SlotNum=1)][Module][Slot(SlotNum=5)][Module][Port(PortNumber=TenGigE0/1/
  5/0)][Phy sicalLayer]}
  </value>
  </param>
  <param name="rs">
  <value>
        <key name="NE Property notification">
        <entry name="register">true</entry>
        <key name="requiredProperties">
        <key name="com.sheer.imo.IPhysicalLayer">
        <entry name="AdminStatusEnum"/>
        <entry name="OperStatusEnum"/>
        </key>
        </key>
        </key>
  </value>
  </param>
</command>
.
```

Retrieving ARP Entries

The following example shows the usage of the BQL **Get** command to retrieve the ARP entries of a network element (core-crs1-p2).

```
<command name="Get">
  <param name="oid">
  <value>
```

```
    {[ManagedElement(Key=core-crs1-
    p2)][LogicalRoot][FWComponentContainer(Type=1)][Rou
    tingEntity][ArpEntity]}
  </value>
  </param>
  <param name="rs">
  <value>
        <key name="com.sheer.imo.keys.IArpEntityOid">
        <entry name="depth">100</entry>
        <entry name="register">true</entry>
        <entry name="cachedResultAcceptable">false</entry>
        <key name="requiredProperties">
        <key name="com.sheer.imo.IMO">
        <entry name="*"/>
        </key>
        </key>
        <key name="requiredAspects">
        <key name="com.sheer.imo.keys.IOid">
        <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
        </key>
        </key>
        </key>
  </value>
  </param>
</command>
.
```

Retrieving All Pseudowires

The following example shows the usage of the BQL **Get** command to retrieve the pseudowires details of a network element (7200 SRD).

```
<?xml version="1.0" encoding="UTF-8"?>
  <command name="Get">
        <param name="oid">
            <value>{[ManagedElement(Key=7200
            SRD)][LogicalRoot][Context(ContextName=Default
            context)][TunnelContainer(TunnelType=1)]}</value>
        </param>
        <param name="rs">
              <value><key name="com.sheer.imo.keys.ITunnelContainerOid">
        <entry name="depth">0</entry>
        <entry name="register">true</entry>
        <entry name="cachedResultAcceptable">false</entry>
        <key name="requiredProperties">
              <key name="com.sheer.imo.technologies.ITunnelEdge">
                    <entry name="*"/>
              </key>
              <key name="com.sheer.imo.technologies.ITunnelContainer">
                    <entry name="*"/>
              </key>
              <key name="com.sheer.imo.IBusinessObject">
                    <entry name="*"/>
              </key>
            </key>
            <key name="requiredAspects">
              <key name="com.sheer.imo.keys.IOid">
                    <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
```

```
            </key>
          </key>
</key></value>
  </param>
</command>
```

Retrieving All VSIs

The following example shows the usage of the BQL **Get** command to retrieve the VSI details of a network element (ASR1K_XE260_V03).

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Get">
  <param name="oid">
<value>{[ManagedElement(Key=ASR1K_XE260_V03)][LogicalRoot][Context(ContextN
ame=Default context)][FWComponentContainer(Type=9)]}</value>
  </param>
  <param name="rs">
        <value><key name="com.sheer.imo.keys.IVSIOid">
              <entry name="depth">100</entry>
              <entry name="register">true</entry>
              <entry name="cachedResultAcceptable">false</entry>
              <key name="requiredProperties">
                    <key name="com.sheer.imo.IMO">
                          <entry name="*"/>
                    </key>
              </key>
              <key name="requiredAspects">
                    <key name="com.sheer.imo.keys.IOid">
                          <entry
                      name="com.sheer.imo.keys.IBusinessObjectOid"/>
                    </key>
              </key>
</key></value>
  </param>
</command>
```

Retrieving a Specific VSI

The following example shows the usage of the BQL **Get** command to retrieve the specific VSI details (VPN ID: 66) of a network element (ASR1K_XE260_V03).

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Get">
  <param name="oid">
<value>{[ManagedElement(Key=ASR1K_XE260_V03)][LogicalRoot][Context(ContextN
ame=Default
context)][FWComponentContainer(Type=9)][VSI(VplsInstanceName=ana1)(VpnId=66
)]}</value>
  </param>
  <param name="rs">
        <value><key name="com.sheer.imo.keys.IVSIOid">
  <entry name="depth">100</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
```

```
        <key name="com.sheer.imo.IMO">
              <entry name="*"/>
        </key>
  </key>
  <key name="requiredAspects">
        <key name="com.sheer.imo.keys.IOid">
              <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
        </key>
  </key>
</key></value>
  </param>
</command>
.
```

Refreshing Inventory Data for an NE

The following example shows the usage of the BQL **Refresh** command to refresh the physical inventory data of a network element.

```
<command name="Refresh">
  <param name="oid">
        <value>{[ManagedElement(Key=PE-209_Sim)]}</value>
  </param>
  <param name="oids">
    <value>{[ManagedElement(Key=PE-
    209_Sim)][PhysicalRoot][Chassis][Slot(SlotNum=1)][Module][P
    ort(PortNumber=FastEthernet1/0)][PhysicalLayer]}</value>
  </param>
</command>
```

Finding the Path Trace

The following example shows the usage of the BQL **GetSNC** command to get the path trace between network elements. Here, you can define the start point (core-crs1-p2) and destination point (layer3data, 172.23.104.11). Here, the path trace is calculated between the port, TenGigE0/1/3/0, and the VLAN identifier, 172.23.104.11.

```
<command name="GetSNC">
  <param name="startingPoint">
<value>{[ManagedElement(Key=core-crs1-
p2)][PhysicalRoot][Chassis][Shelf(ShelfNum=0)][Slot(
SlotNum=1)][Module][Slot(SlotNum=3)][Module][Port(PortNumber=TenGigE0/1/3/0
)][PhysicalLaye r][DataLinkLayer]}</value>
  </param>
  <param name="layer3Data">
        <value>172.23.104.11</value>
  </param>
  <param name="rs">
        <value><key name="path-tool-container">
  <entry name="depth">30000</entry>
  <entry name="register">false</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
        <key name="*">
              <entry name="*"/>
        </key>
  </key>
```

```
<key name="excludedProperties">
      <key name="com.sheer.imo.ILse">
            <entry name="*"/>
      </key>
      <key name="com.sheer.imo.technologies.IAtm">
            <entry name="VcsTable"/>
            <entry name="ContainedCurrentCTPs"/>
            <entry name="ContainingTPs"/>
            <entry name="CrossConnectTable"/>
      </key>
      <key name="com.sheer.imo.technologies.IVcBasedEncapsulation">
            <entry name="ContainedCurrentCTPs"/>
            <entry name="ContainingTPs"/>
            <entry name="Vc"/>
      </key>
      <key name="com.sheer.imo.IRoutingEntity">
            <entry name="*"/>
      </key>
      <key name="com.sheer.imo.IConnectionTerminationPoint">
            <entry name="ContainedCurrentCTPs"/>
            <entry name="ContainingTPs"/>
      </key>
      <key name="com.sheer.imo.IVcCrossConnect">
            <entry name="*"/>
      </key>
      <key name="com.sheer.imo.IBridge">
            <entry name="*"/>
      </key>
      <key name="com.sheer.imo.IVrf">
            <entry name="*"/>
      </key>
      <key name="com.sheer.imo.IPhysicalLayer">
            <entry name="SupportedAlarms"/>
            <entry name="ContainedCurrentCTPs"/>
            <entry name="ContainingTPs"/>
      </key>
      <key name="com.sheer.imo.technologies.IIPInterface">
            <entry name="IpPort"/>
            <entry name="Interfaces"/>
            <entry name="CarEntries"/>
      </key>
      <key name="com.sheer.imo.technologies.IFrameRelay">
            <entry name="VcsTable"/>
            <entry name="ContainedCurrentCTPs"/>
            <entry name="ContainingTPs"/>
            <entry name="CrossConnectTable"/>
      </key>
      <key name="com.sheer.imo.IVcSwitchingEntity">
            <entry name="*"/>
      </key>
      <key name="com.sheer.imo.technologies.IL2TPTunnel">
            <entry name="SessionsTable"/>
      </key>
      <key name="com.sheer.imo.technologies.IMpls">
            <entry name="OutLabels"/>
          <entry name="ContainedCurrentCTPs"/>
          <entry name="MplsTEProperties"/>
```

```
            <entry name="ContainingTPs"/>
            <entry name="InLabels"/>
        </key>
        <key name="com.sheer.imo.technologies.IVlanInterface">
              <entry name="VlanEntries"/>
              <entry name="ContainingTPs"/>
        </key>
  </key>
</key></value>
        </param>
      <param name="simulatedTimestamp">
    <value>-1</value>
  </param>
</command>
```

Attaching a Business Tag

The following example shows the usage of the BQL **Create** command to attach a business tag to a network element (PE-209_Sim).

```
<command name="Create">
  <param name="imobject">
  <value>
        <IBusinessObject>
        <ID
        type="Oid">{[ManagedElement(Key=PE-
        209_Sim)][PhysicalRoot][Chassis][Slot(SlotNum=0
        )][Module][Port(PortNumber=FastEthernet0/0)][BusinessObject]}</ID>
        <EKey type="String">xyz</EKey>
        <Name type="String">Hello World</Name>
        <Notes type="String" />
        <TypeEnum type="Integer">1</TypeEnum>
        </IBusinessObject>
  </value>
  </param>
</command>
```

Editing a Business Tag

The following example shows the usage of the BQL **Update** command to edit a business tag, attached to a network element (PE-209_Sim).

```
<command name="Update">
  <param name="oid">
  <value>{[ManagedElement(Key=PE-
  209_Sim)][PhysicalRoot][Chassis][Slot(SlotNum=0)][Modul
  e][Port(PortNumber=FastEthernet0/0)][BusinessObject]}</value>
  </param>
  <param name="imobjectArr">
  <value>
        <IScalarNotification>
        <ID type="Oid">{[Notification]}</ID>
        <NewIMO type="IBusinessObject">
        <ID
        type="Oid">{[ManagedElement(Key=PE-
        209_Sim)][PhysicalRoot][Chassis][Slot(SlotNum=0
        )][Module][Port(PortNumber=FastEthernet0/0)][BusinessObject]}</ID>
```

```
        <Name type="String">Hello World again</Name>
        </NewIMO>
        <PropertyName type="String">Name</PropertyName>
        </IScalarNotification>
  </value>
  </param>
</command>
```

Finding All Maps

The following example shows the usage of the BQL **Find** command to list all the maps in Prime Network.

```
<command name="Find">
  <param name="imo">
        <value>
              <IMap />
        </value>
  </param>
  <param name="rs">
        <value><key name="FindAllMaps">
  <entry name="depth">100</entry>
  <entry name="register">false</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.IMap">
              <entry name="Name"/>
              <entry name="HierarchyRootOid"/>
        </key>
  </key>
  <key name="excludedProperties">
        <key name="com.sheer.imo.IMap">
              <entry name="Links"/>
        </key>
  </key>
</key></value>
  </param>
</command>
```

Getting Map Details

The following example shows the usage of the BQL **Get** command to retrieve the details of a map (21001).

```
<command name="Get">
  <param name="oid">
        <value>{[Map(Id=21001)]}</value>
  </param>
  <param name="rs">
        <value><key name="GetMapDetails">
  <entry name="depth">100</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">true</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.IHierarchyNode">
              <entry name="Name"/>
         <entry name="ManagedParent"/>
```

```
      <entry name="Map"/>
      <entry name="Children"/>
      <entry name="Leaf"/>
      <entry name="ContainedObjectOid"/>
    </key>
    <key name="com.sheer.imo.IReconciliationAspect">
          <entry name="TypeEnum"/>
    </key>
    <key name="com.sheer.imo.newalarm.ITicket">
          <entry name="Source"/>
          <entry name="EventCount"/>
          <entry name="LatestState"/>
          <entry name="AggregatedAckStateEnum"/>
          <entry name="AutoCleared"/>
          <entry name="Archived"/>
          <entry name="LastModificationTime"/>
          <entry name="AggregatedSeverityEnum"/>
          <entry name="ReductionCount"/>
          <entry name="DuplicationCount"/>
          <entry name="AffectedDevicesCount"/>
          <entry name="AlarmCount"/>
    </key>
  <key name="com.sheer.imo.ILink">
    <entry name="BiDirectional"/>
    <entry name="ConnectionInformation"/>
    <entry name="LinkTypeEnum"/>
  </key>
  <key name="com.sheer.imo.IVlanEntryReferencedStpAspect">
    <entry name="StpPortInfo"/>
  </key>
  <key name="com.sheer.imo.INE">
    <entry name="ScriptMetadataOids"/>
  </key>
  <key name="com.sheer.imo.IHierarchyChildrenAspect">
    <entry name="Children"/>
  </key>
  <key name="com.sheer.imo.IBusinessObject">
    <entry name="Name"/>
    <entry name="Notes"/>
    <entry name="TypeEnum"/>
    <entry name="EKey"/>
  </key>
  <key name="com.sheer.imo.IBridge">
    <entry name="StpInstanceInfo"/>
    <entry name="ScriptMetadataOids"/>
  </key>
  <key name="com.sheer.imo.IMapDataAspect">
    <entry name="LinksTypeFilter"/>
  </key>
  <key name="com.sheer.imo.IContainedImo">
          <entry name="ContainedObjectOid"/>
  </key>
  <key name="com.sheer.imo.IReferencedImosAspect">
    <entry name="Efd"/>
  </key>
  <key name="com.sheer.imo.IReferencedPortInfoAspect">
    <entry name="*"/>
```

```
</key>
<key name="com.sheer.imo.IReferencedEndPointsAspect">
  <entry name="ZEndPoint"/>
  <entry name="AEndPoint"/>
</key>
<key name="com.sheer.imo.newalarm.ITicketListAspect">
  <entry name="Tickets"/>
</key>
<key name="com.sheer.imo.ITopologicalLink">
  <entry name="MaintenanceMode"/>
  <entry name="DetectionTypeEnum"/>
  <entry name="BiDirectional"/>
  <entry name="ConnectionInformation"/>
  <entry name="LinkTypeEnum"/>
</key>
<key name="com.sheer.imo.IStpPortsAspect">
  <entry name="*"/>
</key>
<key name="com.sheer.imo.IContainedHierarchyNode">
  <entry name="HierarchyRootOid"/>
</key>
<key name="com.sheer.imo.IMapAspect">
  <entry name="XCoordinate"/>
  <entry name="YCoordinate"/>
  <entry name="MapOid"/>
  <entry name="Height"/>
  <entry name="Background"/>
  <entry name="SubGraph"/>
  <entry name="Width"/>
</key>
<key name="com.sheer.imo.technologies.IIPInterface">
  <entry name="Address"/>
</key>
<key name="com.sheer.imo.IEthernetFlowDomain">
  <entry name="Name"/>
</key>
<key name="com.sheer.imo.IBusinessElement">
  <entry name="*"/>
</key>
<key name="com.sheer.imo.IReferencedBridgeAspect">
  <entry name="Bridge"/>
</key>
<key name="com.sheer.imo.IBusinessLink">
  <entry name="ZEndPoint"/>
  <entry name="AEndPoint"/>
  <entry name="BiDirectional"/>
  <entry name="LinkTypeEnum"/>
</key>
<key name="com.sheer.imo.IAlarmBusinessObject">
  <entry name="*"/>
</key>
<key name="com.sheer.imo.IRepAspect">
  <entry name="*"/>
</key>
<key name="com.sheer.imo.technologies.IStpPortInfo">
  <entry name="StpPortState"/>
  <entry name="StpPortRole"/>
```

```
      <entry name="ScriptMetadataOids"/>
    </key>
    <key name="com.sheer.imo.IReferencedStpInstanceInfoAspect">
      <entry name="StpInstanceInfo"/>
    </key>
    <key name="com.sheer.imo.IVrf">
      <entry name="ScriptMetadataOids"/>
  </key>
  <key name="com.sheer.imo.technologies.IVlanEntry">
      <entry name="StpPortInfo"/>
      <entry name="ScriptMetadataOids"/>
    </key>
    <key name="com.sheer.imo.IReferencedStpPortsInfoAspect">
      <entry name="ASideStpPortInfo"/>
      <entry name="ZSideStpPortInfo"/>
    </key>
    <key name="com.sheer.imo.IMapLinksAspect">
      <entry name="Links"/>
    </key>
    <key name="com.sheer.imo.technologies.IStpInstanceInfo">
      <entry name="IsRoot"/>
      <entry name="ScriptMetadataOids"/>
    </key>
    <key name="com.sheer.imo.ISeverityAspect">
      <entry name="*"/>
    </key>
    <key name="com.sheer.imo.ICapacityExceededAspect">
      <entry name="*"/>
    </key>
    <key name="com.sheer.imo.IMap">
      <entry name="Name"/>
      <entry name="Links"/>
      <entry name="HierarchyRootOid"/>
    </key>
    <key name="com.sheer.imo.technologies.vendors.cisco.IREPPortInfo">
      <entry name="SegmentId"/>
      <entry name="PortRole"/>
      <entry name="BlockedVlans"/>
      <entry name="PortType"/>
      <entry name="OperPortStatus"/>
    </key>
    <key name="com.sheer.imo.IEthFlowPointReferencedNEAspect">
      <entry name="NetworkElement"/>
    </key>
    <key name="com.sheer.imo.IManagedElement">
      <entry name="SysLocation"/>
      <entry name="DeviceName"/>
      <entry name="ElementType"/>
      <entry name="IP"/>
      <entry name="SysUpTime"/>
      <entry name="VendorEnum"/>
      <entry name="ElementTypeKey"/>
      <entry name="ScriptMetadataOids"/>
      <entry name="CommunicationStateEnum"/>
      <entry name="ElementCategoryEnum"/>
      <entry name="SoftwareVersion"/>
      <entry name="SysDescription"/>
```

```
    <entry name="InvestigationStateEnum"/>
    <entry name="SysName"/>
  </key>
  </key>
  <key name="requiredAspects">
    <key name="com.sheer.imo.keys.IManagedElementOid">
          <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
          <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
          <entry name="com.sheer.imo.keys.ISeverityAspectOid"/>
    </key>
    <key name="com.sheer.imo.keys.IMapOid">
          <entry
          name="com.sheer.imo.keys.IContainedHierarchyNodeOid"/>
          <entry name="com.sheer.imo.keys.IMapDataAspectOid"/>
          <entry name="com.sheer.imo.keys.IMapLinksAspectOid"/>
          <entry
          name="com.sheer.imo.keys.ICapacityExceededAspectOid"/>
    </key>
    <key name="com.sheer.imo.keys.IBusinessElementOid">
          <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
    </key>
    <key name="com.sheer.imo.keys.IVlanEntryOid">
          <entry name="com.sheer.imo.keys.IReconciliationAspectOid"/>
          <entry
          name="com.sheer.imo.keys.IVlanEntryReferencedStpAspectOid"/
          >
          <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
          <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
          <entry name="com.sheer.imo.keys.ISeverityAspectOid"/>
    </key>
    <key name="com.sheer.imo.keys.ILinkOid">
          <entry name="com.sheer.imo.keys.IRepAspectOid"/>
          <entry name="com.sheer.imo.keys.IStpPortsAspectOid"/>
          <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
          <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
          <entry name="com.sheer.imo.keys.ISeverityAspectOid"/>
    </key>
    <key name="com.sheer.imo.keys.IBridgeOid">
          <entry name="com.sheer.imo.keys.IReconciliationAspectOid"/>
          <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
          <entry
          name="com.sheer.imo.keys.IReferencedStpInstanceInfoAspectOi
          d"/>
          <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
          <entry name="com.sheer.imo.keys.ISeverityAspectOid"/>
    </key>
    <key name="com.sheer.imo.keys.INewAlarmOid">
          <entry name="com.sheer.imo.keys.IAlarmBusinessObjectOid"/>
    </key>
    <key name="com.sheer.imo.keys.IHierarchyNodeOid">
          <entry name="com.sheer.imo.keys.IMapAspectOid"/>
          <entry
          name="com.sheer.imo.keys.IHierarchyChildrenAspectOid"/>
          <entry name="com.sheer.imo.keys.IContainedImoOid"/>
    </key>
    <key name="com.sheer.imo.keys.IEthFlowPointOid">
```

```
            <entry
            name="com.sheer.imo.keys.IEthFlowPointReferencedNEAspectOid
            "/>
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
      </key>
      <key name="com.sheer.imo.keys.ISwitchingEntityOid">
            <entry
            name="com.sheer.imo.keys.IReferencedBridgeAspectOid"/>
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
      </key>
      <key name="com.sheer.imo.keys.INetworkVlanOid">
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
            <entry name="com.sheer.imo.keys.IReferencedImosAspectOid"/>
      </key>
      <key name="com.sheer.imo.keys.IRepAspectOid">
            <entry
          name="com.sheer.imo.keys.IReferencedPortInfoAspectOid"/>
      </key>
      <key name="com.sheer.imo.keys.IStpPortsAspectOid">
        <entry
      name="com.sheer.imo.keys.IReferencedStpPortsInfoAspectOid"/>
      </key>
      <key name="com.sheer.imo.keys.IBusinessLinkOid">
            <entry
          name="com.sheer.imo.keys.IReferencedEndPointsAspectOid"/>
      </key>
      <key name="com.sheer.imo.keys.INEOid">
            <entry name="com.sheer.imo.keys.IReconciliationAspectOid"/>
            <entry name="com.sheer.imo.keys.ITicketListAspectOid"/>
            <entry name="com.sheer.imo.keys.IBusinessObjectOid"/>
            <entry name="com.sheer.imo.keys.ISeverityAspectOid"/>
      </key>
    </key>
</key></value>
  </param>
</command>
```

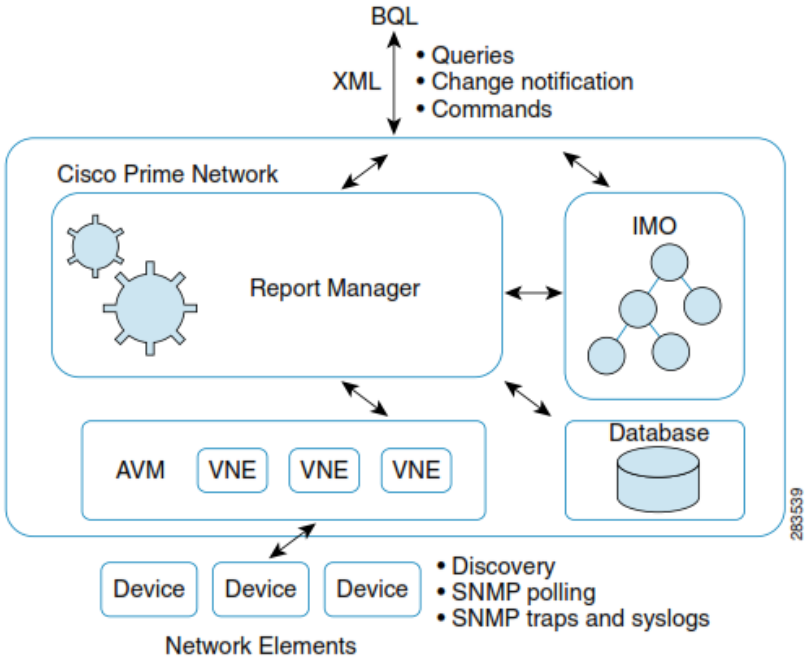
5.5 Generating Standard Reports Using BQL

The Report Manager enables a centralized and flexible interface to define, run, and view different reports of the data that are managed by Cisco Prime Network. The Report Manager retrieves data from the Prime Network database and from the managed VNEs. You can save the generated reports in any of the following formats: PDF, CSV, HTML, XLS, and XML. In addition to a variety of standard reports for events and inventory, you can define reports based on the available standard reports templates.

Figure 5-7 shows a high-level overview of how BQL can be used to manage standard reports.

Note: BQL cannot be used for the new operations reports introduced in Prime Network 4.0.

Figure 5-7 Managing Standard Reports using BQL



Prime Network uses Oracle database to persist any required data. You can define the purge and maximum storage policies for the generated reports. By default, the reports that are older than 90 days are removed. You can also specify whether users can create public reports.

Additional Reading

- Review *Cisco Prime Network 4.2.2 Administrator Guide* to understand the Prime Network user roles and scopes.
- Review *Cisco Prime Network 4.2.2 User Guide* to understand the different types of reports.
- See the *Cisco Prime Network Information Model* Javadoc to understand the IMO for reports. This document is available on the Prime Network Technology Center website. You must have a Cisco.com account with partner level access, or you must be a Prime Network licensee to access this website.

5.5.1 Report Categories

The following standard reports are supported in Prime Network:

- Events—See Events Reports, page 148.
- Inventory—See Inventory Reports, page 153.
- Network Services—See Network Service Reports, page 155.

Events Reports

Table 5-6 describes the standard events report types provided by Prime Network.

Table 5-6 Standard Events Report Types

Report Name	Description	BQL Input Report Parameter (IReportParameter)
Fault DB vs. Event Archive Statistics	For each day in the specified time period, the number of each of the following items in the alarm database and the Prime Network Event Collector (EC): • Syslogs • Traps • Tickets • Correlated events • Uncorrelated events • Nonnetwork events • Network-originated events • Network-originated and service events	Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details.
Daily Average and Peak	For each day of the specified time period, the peak number and average rate of syslogs and traps for each of the following time periods: • Second • Ten seconds • Minute • Hour • Day	Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details.

Detailed Event Count (By Device)	For each device, the following information for the specified time period:
	• Number of syslogs for each severity, including
	  • Syslog type
	  • Number of each syslog type
	• Number of traps for each severity, including:
	  • Trap type
	  • Number of each trap type
		• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details.
		• SelectedDevices. See Specifying Network Elements page 161 for details.

Report Name	Description	BQL Input Report Parameter (IReportParameter)

	• Number of tickets, including:
	  • Ticket type
	  • Number of each ticket type	

Detailed Syslogs	For each device, the following information from the event archive for the specified time period: • Date and time of each syslog, in ascending order • Raw syslog The maximum number of syslogs retrieved for this report is 250,000. You can run this report only on the event archive.	• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details. • SelectedDevices. See Specifying Network Elements page 161 for details.

Detailed Traps	For each device, the following information for the specified time period:
	• IP address
	• Time of trap
	• SNMP version
	• Generic or device-specific trap OID
	• Detailed trap description
	The maximum number of traps retrieved for this report is 250,000. You can run this report only on the event archive.
		• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details.
		• SelectedDevices. See Specifying Network Elements page 161 for details.
		• version—Specify the SNMP version to include in the report: 1, 2, or 3. See Generating Detailed Traps Report, page 173 for example.
		• vgeneric—Specify the generic traps to include in the report:
		  • coldStart
		  • 1—warmStart
		  • 2—linkDown
		  • 3—linkUp
		  • 4—authenticationFailure
		  • 5—egpNeighborLoss
		  • 6—enterpriseSpecific
		See Generating Detailed Traps Report, page 173 for example.
		• VSpecific—If you select generic type 6, you must pass the OIDs (comma separated). See Generating Detailed Traps Report, page 173 for example.

Report Name	Description	BQL Input Report Parameter (IReportParameter)
Devices with the Most Events (By Severity)	For the specified number of devices with the most events, the following information for each device for the specified time period: • Severity of the events associated with the device, sorted by severity. • Number of events for each severity. A pie chart presents the information by device and percentage in a graphical format.	• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details. • SelectedDevices. See Specifying Network Elements page 161 for details. • numberOfDevices—Number of items to be displayed in the generated report.
Devices with the Most Events (By Type)	For the specified number of devices with the most events, the following information for each device for the specified time period: • Type of events associated with the device • Number of events received for each event type A pie chart presents the information by device and percentage in a graphical format.	• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details. • SelectedDevices. See Specifying Network Elements page 161 for details. • numberOfDevices—Number of items to be displayed in the generated report.
Devices with the Most Syslogs	For the specified number of devices with the most syslogs, the number of syslog messages for each device for the specified time period. You can run this report on the Prime Network alarm database or the event archive. A pie chart presents the information by device and percentage in a graphical format.	• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details. • SelectedDevices. See Specifying Network Elements page 161 for details. • numberOfDevices—Number of items to be displayed in the generated report. See Generating Devices with the Most SyslogsGenerating Devices with the Most Syslogs, page 174 for example.
Devices with the Most Traps	For the specified number of devices with the most traps, the number of traps associated with each device for the specified time period.	• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details.

Report Name	Description	BQL Input Report Parameter (IReportParameter)
	You can run this report on the Prime Network database or the event archive. A pie chart presents the information by device and percentage in a graphical format.	• SelectedDevices. See Specifying Network Elements page 161 for details. • numberOfDevices—Number of items to be displayed in the generated report.
Most Common Daily Events	For each day in the specified time period: • The specified number of most common syslogs, traps, tickets, and service events • The number of each type of syslog, trap, ticket, and service event If selected, a pie chart that presents the events by percentage in a graphical format	• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details. • SelectedDevices. See Specifying Network Elements page 161 for details. numberOfMessages—Number of items to be displayed in the generated report. See Generating Most Commonly Daily Events, page 176 for example. • Show Charts—To view pie charts in the report with the standard numerical output. See Generating Most Commonly Daily Events, page 176 for example.
Most Common Syslogs	Most common syslog messages and the number of each for the specified time period and devices. A pie chart presents the information by syslog message and percentage in a graphical format.	• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details. • SelectedDevices. See Specifying Network Elements page 161 for details. • numberOfMessages—Number of items to be displayed in the generated report. See Generating Most Commonly Daily Events, page 176 for example.
Syslog Count	Number of syslog messages by type for the specified time period with the times of the first and last occurrences.	• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details.

Report Name	Description	BQL Input Report Parameter (IReportParameter)
	A pie chart presents the information by syslog message and percentage in a graphical format.	• SelectedDevices. See Specifying Network Elements page 161 for details.
Syslog Count (By Device)	For each device, the type and number of each syslog message and the times of the first and last occurrences for each type. A pie chart presents the information by device and percentage in a graphical format.	• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details. • SelectedDevices. See Specifying Network Elements page 161 for details.

Syslog Trend (By Severity)	For the specified devices, the trend of specified syslog messages in graph format:
	• By priority
	• For the specified time period
	• At the specified intervals
	You can run this report on the Prime Network alarm database.
		• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details.
		• SelectedDevices. See Specifying Network Elements page 161 for details.
		• IntervalUnit—Specify the interval in Seconds, Minutes, Hours, and Days. The IntervalUnit is the units displayed in the chart x-axis (the y-axis includes the syslog counts).
		• SelectedSeverity
		  • 0—Critical
		  • 1—Major
		  • 2—Minor
		  • 3—Warning
		  • 4—Cleared
		  • 5—Information
		  • 6—Indeterminate
		• SelectedSyslogMsgs—Refer to the send-alarm-msg-util.xml file under NETWORKHOME/Main/registry for syslog message. In this file use the value specified in the short-description entry name.
		<key name="ACE-727001-syslog">

Report Name	Description	BQL Input Report Parameter (IReportParameter)
		<keyname="ACE-727001-syslog"> <entryname="default">send-alarm-msg-util/templates/syslogs-template</entry> <entryname="alarm-type">9045</entry> <entry name="severity">MAJOR</entry> <entryname="short-description">Peer IP address is not reachable syslog</entry> </key>

Inventory Reports

Table 5-7 describes the standard inventory report types provided by Prime Network.

Table 5-7 Standard Inventory Report Types

Report Name	Description	BQL Input Report Parameter (IReportParameter)
Hardware Summary (Detailed)	For each device included in the report: • IP address • Device series • Element type You can view other hardware information for each device by selecting the required items from the available list as given below: • Chassis—chassis description, chassis serial number, shelf description, shelf serial number, and shelf status • Module—module name, sub module name, module status, hardware type, and hardware version	• SelectedDevices. See Specifying Network Elements page 161 for details. • SelectedColumns—Specify a string to view chassis, module, or port related hardware information.

Report Name	Description	BQL Input Report Parameter (IReportParameter)
	• Port—port location, port type, porting sending alarm, port alias, port status, port managed, PID, and pluggable type serial number.	
Hardware Summary (By Selected Property)	For each device included in the report: • IP address • System name • Serial number • Element type • Device series • Vendor • Product • Chassis You can group the report contents by vendor, product, device series, element type, system name, or chassis and specify part or whole of the selected entity, if required.	• SelectedDevices. See Specifying Network Elements page 161 for details. • GroupFilter—Specify a string to group the report contents by vendor, product, device series, element type, system name, or chassis. • Filter—(Optional) Specify a sting to filter the GroupFilter parameter values by using a specific value.
Software Summary (By Device)	For each device included in the report: • Device name • Element type • IP address • Serial number • Software version on the device • Name of image file	• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details. • SelectedDevices. See Specifying Network Elements page 161 for details.
Software Summary (By Version)	For each software version included in the report: • Number of devices running the version • Device names • Element types • Device IP address • Device serial number • Name of image file	• Todate, FromDate, and IsAllDates. See Specifying Date Format, page 158 for details. • SelectedDevices See Specifying Network Elements page 161 for details.

Report Name	Description	BQL Input Report Parameter (IReportParameter)
Modules Summary (By Type)	For each device filtered by module type: • IP address • Module serial number • Module hardware version • Module software version You can filter the report contents by specifying part or whole of the module type.	• SelectedDevices See Specifying Network Elements page 161 for details. • Filter—(Optional) Specify a sting to filter the module types by using a specific value.

Network Service Reports

Table 5-7 describes the standard network service report types provided by Prime Network.

Table 5-7 Standard Network Service Report Types

Report Name	Description	BQL Input Report Parameter (IReportParameter)
AToM Detailed	For each pseudowire in the report: • Pseudowire name • Pseudowire type • Business tag assigned to the pseudowire • Maps that contain the pseudowire • Pseudowire endpoints • Switching entities • Ethernet flow points • Connection termination points (TPs) You can filter report content by specifying part or all of the: • Pseudowire name • Pseudowire type • Business tag • Map name	The following parameters are optional: • PseudowireType—Specify a string to include the pseudowire type. • BusinessTag—Specify a string to include the business tag that is attached to pseudowire. • PseudowireName—Specify a string to include the pseudowire name. • MapList—Specify the maps.
AToM Summary	For each pseudowire in the report: • Pseudowire name	The following parameters are optional:

Report Name	Description	BQL Input Report Parameter (IReportParameter)
	• Pseudowire type • Business tag assigned to the pseudowire • Maps that contain the pseudowire You can filter the report content by specifying part or all of the: • Pseudowire name • Pseudowire type • Business tag • Map name	• PseudowireType—Specify a string to include the pseudowire type. • BusinessTag—Specify a string to include the business tag that is attached to pseudowire. • PseudowireName—Specify a string to include the pseudowire name. • MapList—Specify the maps.
Ethernet Service Detailed	For each Ethernet service in the report: • Ethernet service or Layer 2 VPN name • Business tag assigned to the Ethernet service or Layer 2 VPN instance • EVC name • Business tag assigned to the EVC • Maps that contain the Ethernet service or Layer 2 VPN • Edge EFPs associated with the EVC or Layer 2 VPN • EFD fragment names • EFD fragment types You can filter report content by specifying part or all of the: • Ethernet service name • EVC name • Ethernet service business tag • EVC business tag • Map name	The following parameters are optional: • EVCBusinessTag—Specify a string to include the business tag that is attached to Ethernet Virtual Connection (EVC). • EthernetServiceBusinessTag—Specify a string to include the business tag that is attached to Ethernet Service. • EthernetServiceName—Specify a string to include the Ethernet Service name. • EVCName—Specify a string to include the EVC name. • MapList—Specify the maps.
Ethernet Service Summary	For each Ethernet service in the report:	The following parameters are optional:

Report Name	Description	BQL Input Report Parameter (IReportParameter)
	• Ethernet service or Layer 2 VPN name • Business tag assigned to the Ethernet service or Layer 2 VPN instance • EVC name • Business tag assigned to the EVC • Maps that contain the Ethernet service or Layer 2 VPN You can filter report content by specifying part or all of the: • Ethernet service name • EVC name • Ethernet service business tag • EVC business tag • Map name	• EVCBusinessTag—Specify a string to include the business tag that is attached to Ethernet Virtual Connection (EVC). • EthernetServiceBusinessTag—Specify a string to include the business tag that is attached to Ethernet Service. • EthernetServiceName—Specify a string to include the Ethernet Service name. • EVCName—Specify a string to include the EVC name. • MapList—Specify the maps.
VPLS Detailed	For each Virtual Private LAN Service (VPLS) or Hierarchical VPLS (H-VPLS) instance in the report: • VPLS or H-VPLS name • Business tag associated with the VPLS or H-VPLS instance • Maps that contain the VPLS or H-VPLS • instance • VPLS forwarders that represent the de- vice-level VPLS switching entities • Access pseudowire endpoints • Access EFPs • Core pseudowires You can filter report content by specifying part or all of the: • VPLS or H-VPLS name • Business tag • Map name	• vplsName—Specify a string to include the VPLS name. • BusinessTag—Specify a string to include the business tag that is attached to VPLS. • MapList—Specify the maps.

Report Name	Description	BQL Input Report Parameter (IReportParameter)
VPLS Summary	For each VPLS or H-VPLS instance in the report: • VPLS or H-VPLS name • Business tag assigned to the VPLS or • H-VPLS instance • Maps that contain the VPLS or H-VPLS • instance You can filter report content by specifying part or all of the: • VPLS or H-VPLS name • Business tag • Map name	• vplsName—Specify a string to include the VPLS name. • BusinessTag—Specify a string to include the business tag that is attached to VPLS. • MapList—Specify the maps.

5.5.1.1 Specifying Date Format

In BQL, the date must be defined in the UNIX timestamp format (in milliseconds). You can use the online timestamp convertor tool available at this URL: http://www.epochconverter.com/. Most of the online timestamp convertor tool converts the human timestamp into seconds. You must append 000 before parsing in BQL.

Date Format for Generating Reports

While generating the reports, you can specify dates in the following formats:

- **Last**—The amount of time prior to the current date and time.
- **From Date and To Date**—The date range for the report.

For example, to generate a report for the last 5 days or to generate a report from 6 May 2010 to 11 May 2010, the following BQL parameters are passed:

- Todate = 1273578242051, specifies the date as Tue, 11 May 2010 11:44:02.
- FromDate = 1273146242051, specifies the date as Thu, 6 May 2010 11:44:02 UTC.
- IsAllDates = false, specifies this parameter is not used.

```
<value>
  <IReportParameter>
        <Name type="String">ToDate</Name>
              <Value type="java.lang.String_Array">
                    <java.lang.String>1273578242051</java.lang.String>
               </Value>
  </IReportParameter>
```

```
</value>
<value>
  <IReportParameter>
        <Name type="String">FromDate</Name>
              <Value type="java.lang.String_Array">
                    <java.lang.String>1273146242051</java.lang.String>
              </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">IsAllDates</Name>
              <Value type="java.lang.String_Array">
                    <java.lang.String>false</java.lang.String>
              </Value>
  </IReportParameter>
</value>
```

Date Format for Creating Reports

While defining a new report, you have to specify the date in the following format (see Example 1: Date Format for Date Range and Example 2: Date Format for Specific Dates:

```
<value>
  <IReportParameter>
        <Name type="String">Date Group - OrderedDateRange</Name>
              <Value type="java.lang.String_Array">
                    <java.lang.String>Type,First_Parameter,Second_Paramet
              er</java.lang.String>
              </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">IsAllDates</Name>
              <Value type="java.lang.String_Array">
                    <java.lang.String>false</java.lang.String>
              </Value>
  </IReportParameter>
</value>
```

Where,

- *Type* specifies the date selection format:
  - 1—All Dates. Not supported in this release.
  - 2—To specify date range; for example: 1 day, 120 minutes, and so on.
  - 3—To specify specific dates; for example: from 11 May 2010 to 11 June 2010.
- *First_Parameter and Second_Parameter* specifies the from and to date if you are using specific dates. If specific date range is used, *First_Parameter* denotes the value and *Second_Parameter* denotes the unit; Where unit is:
  - 0—Seconds
  - 1—Minutes
  - 2—Hours

- 3—Days
- 4—Weeks

Example 1: Date Format for Date Range

For example, to generate report for the last 5 days, the following BQL parameters are passed:

```
<IReportParameter>
  <Name type="String">Date Group - OrderedDateRange</Name>
  <Value type="java.lang.String_Array">
        <java.lang.String>2,5,3</java.lang.String>
  </Value>
</IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">IsAllDates</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>false</java.lang.String>
        </Value>
  </IReportParameter>
```

In the above example:

- 2—Specific date range is passed
- 3—Date range is specified in terms of days.
- 5—Number of days for which the report data is displayed.

Example 2: Date Format for Specific Dates

For example, to generate report from 1 May 2010 to 12 May 2010, the following BQL parameters are passed:

```
<IReportParameter>
  <Name type="String">Date Group - OrderedDateRange</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>3,1273430632000,1273776232000</java.lang.
              String>
        </Value>
     </IReportParameter>
   </value>
        <value>
              <IReportParameter>
                    <Name type="String">IsAllDates</Name>
                          <Value type="java.lang.String_Array">
                                <java.lang.String>false</java.lang.String
                              >
                          </Value>
              </IReportParameter>
```

In the above example:

- 3—Specific dates are passed.
- 1273430632000—From date (Sun, 9 May 2010 18:43:52 UTC).
- 1273776232000—To date (Thu, 13 May 2010 18:43:52 UTC).

5.5.1.2 Specifying Network Elements

You have to specify the VNE key individually for every VNE as part of the SelectedDevices parameter when you generate the report. In the following example, ASR10k and C2900 are the VNE keys for which the report is generated.

SelectedDevices Format while Generating Reports (BQL Command: RunReportTypeCommand)

```
<value>
  <IReportParameter>
        <Name type="String">SelectedDevices</Name>
        <Value type="java.lang.String_Array">
<java.lang.String>'{[ManagedElement(Key=ASR10k_SBC)]}'</java.lang.String>
<java.lang.String>'{[ManagedElement(Key=C2900)]}'</java.lang.String>
        </Value>
  </IReportParameter>
</value>
```

SelectedDevices Format while Creating Reports (BQL Command: SaveReportTypeCommand)

```
<value>
        <IReportParameter>
              <Name type="String">SelectedDevices</Name>
              <Value type="java.lang.String_Array">
                    <java.lang.String>ASR10k_SBC</java.lang.String>
                    <java.lang.String>C2900</java.lang.String>
              </Value>
        </IReportParameter>
</value>
```

5.5.1.3 Viewing Reports

You have to use the browser to view the report. Once the report is generated, use the following URL to view the report:

https://PrimeNetwork_Gateway_IP_Address:https-port/ana/services/reports/viewReport?id=report-id%26format=Report-Format

Where,

- *PrimeNetwork_Gateway_IP_Address* = IP address of the Prime Network gateway.
- *https-port* = Port for HTTPS connections to Prime Network gateway. By default it is 6081. You can get the HTTPS port information in the mmvm.xml registry file (https-port).

report-id = Report ID of the generated report. This ID is displayed after report is generated. See Identifying Report ID, page 161 for details.

- *Report-Format* = Format of the report. The supported formats are PDF, CSV, HTML, XLS, and XML.

Identifying Report ID

After generating the report using BQL, the following output is displayed. In this example, the report ID is 9.

```
<?xml version="1.0" encoding="UTF-8"?>
<IReport type=”IReport” instance_id=”1”>
  <ID type="Oid">{[Report(Id=9)]}</ID>
  <CreationTime type="java.util.Date">Tue May 11 17:20:26 IST
  2010</CreationTime>
  <DataOid type="">Null</DataOid>
  <DataSourceEnum type="Integer">0</DataSourceEnum>
  <Description type="String">BQL Sample- Prime Network vs Event
  Archive</Description>
  <Name type="String">BQL Sample- Prime Network vs Event Archive</Name>
  <Pub type="Boolean">false</Pub>
  <StateEnum type="Integer">0</StateEnum>
  <Type type="String">Prime Network vs. Event Archive Statistics</Type>
  <URI type="String">/reportfw/rptdocument/BQL Sample- Prime Network vs
  Event
Archive-9.rptdocument</URI>
  <Username type="String">root</Username>
</IReport>
```

To view this report in PDF format on the browser, you must enter the following as the URL:
*https://PrimeNetwork_Gateway_IP_Address:6081/ana/services/reports/viewReport?id=**9**%26format=P DF*

5.5.2 Scheduling Reports and Managing Scheduled Jobs

You can schedule a report to run immediately or at a later point in time. The following example shows a BQL command for generating a Modules Summary report with the scheduling option:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Create">
  <param name="imobject">
        <value>
              <scheduler.IJob type="scheduler.IJob" instance_id="0">
                    <ID type="Oid">{[Job(Name=MOD_1)]}</ID>
                    <Comment type="String" />
                    <ScheduleTypeEnum type="Integer">0</ScheduleTypeEnum>
                    <EndTime type="java.util.Date">Fri May 18 13:52:49 IST
                    2012</EndTime>
                    <StartTime type="java.util.Date">Fri May 18 13:52:49
                    IST 2012</StartTime>
                    <CID type="String"><?xml version="1.0" encoding="UTF-
                    8"?>
<command name="RunReportTypeCommand">
  <param name="name">
        <value>MOD_1</value>
  </param>
  <param name="description">
        <value>Module summary</value>
  </param>
  <param name="type">
        <value>Modules Summary (By Type)</value>
  </param>
  <param name="source">
```

```
        <value>2</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="values">
        <value>
              <IReportParameter type="IReportParameter" instance_id="0">
                    <Name type="String">Filter</Name>
                    <Value type="java.lang.String_Array">
                          <java.lang.String />
                    </Value>
              </IReportParameter>
        </value>
<value>
  <IReportParameter type="IReportParameter" instance_id="0">
        <Name type="String">SelectedDevices</Name>
        <Value type="java.lang.String_Array">
          <java.lang.String>'{[ManagedElement(Key=7600ACE)]}'</java.lang.
        String>
<java.lang.String>'{[ManagedElement(Key=10.56.101.74)]}'</java.lang.String
>
<java.lang.String>'{[ManagedElement(Key=10.56.101.79)]}'</java.lang.String
>
<java.lang.String>'{[ManagedElement(Key=CPT-224)]}'</java.lang.String>

<java.lang.String>'{[ManagedElement(Key=10.56.101.184)]}'</java.lang.Strin
g>
  <java.lang.String>'{[ManagedElement(Key=ucs1)]}'</java.lang.String>
  <java.lang.String>'{[ManagedElement(Key=ucs2)]}'</java.lang.String>
  <java.lang.String>'{[ManagedElement(Key=MWR2941)]}'</java.lang.String>
  <java.lang.String>'{[ManagedElement(Key=NEXUS5K)]}'</java.lang.String>
  <java.lang.String>'{[ManagedElement(Key=p1)]}'</java.lang.String>
  <java.lang.String>'{[ManagedElement(Key=c7-sw8)]}'</java.lang.String>
  <java.lang.String>'{[ManagedElement(Key=p2)]}'</java.lang.String>
<java.lang.String>'{[ManagedElement(Key=c4-npe1-76)]}'</java.lang.String>
<java.lang.String>'{[ManagedElement(Key=10.56.22.105)]}'</java.lang.String
>

<java.lang.String>'{[ManagedElement(Key=10.56.101.163)]}'</java.lang.Strin
g>

<java.lang.String>'{[ManagedElement(Key=10.56.101.183)]}'</java.lang.Strin
g>

<java.lang.String>'{[ManagedElement(Key=c1-npe1-76)]}'</java.lang.String>

<java.lang.String>'{[ManagedElement(Key=C9-AGG20)]}'</java.lang.String>
<java.lang.String>'{[ManagedElement(Key=p3)]}'</java.lang.String>

<java.lang.String>'{[ManagedElement(Key=c9-npe1-9K)]}'</java.lang.String>

<java.lang.String>'{[ManagedElement(Key=c2-npe1-crs)]}'</java.lang.String>
<java.lang.String>'{[ManagedElement(Key=CPT224)]}'</java.lang.String>

<java.lang.String>'{[ManagedElement(Key=c7-npe1-76)]}'</java.lang.String>
<java.lang.String>'{[ManagedElement(Key=2960)]}'</java.lang.String>
```

```
<java.lang.String>'{[ManagedElement(Key=140.1.1.1)]}'</java.lang.String>

<java.lang.String>'{[ManagedElement(Key=10.105.58.252)]}'</java.lang.Strin
g>
                    </Value>
                </IReportParameter>
            </value>
            <value />
        </param>
        <param name="data">
              <value />
        </param>
</command></CID>
              </scheduler.IJob>
            </value>
        </param>
</command>
.
```

You can view the jobs (report or command jobs) that are scheduled to run at a later point in time, by using the following BQL command:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Find">
      <param name="imo">
        <value>
              <scheduler.IJob type="scheduler.IJob" instance_id="0">
                    <ID type="Oid">{[Job]}</ID>
              </scheduler.IJob>
            </value>
        </param>
        <param name="rs">
              <value><key name="">
        <entry name="depth">0</entry>
        <entry name="register">false</entry>
        <entry name="cachedResultAcceptable">false</entry>
        <entry name="getFromSnapshot">true</entry>
        <key name="requiredProperties">
              <key name="com.sheer.imo.scheduler.IJobRun">
                    <entry name="WorkStatusEnum"/>
                    <entry name="StartTime"/>
                    <entry name="ResultStatusEnum"/>
              </key>
              <key name="com.sheer.imo.scheduler.IJob">
                    <entry name="Owner"/>
                    <entry name="Comment"/>
                    <entry name="ScheduleTypeEnum"/>
                    <entry name="StateEnum"/>
                    <entry name="LastJobRun"/>
                    <entry name="CID"/>
                    <entry name="DayOfWeek"/>
                    <entry name="RepeatsInterval"/>
                    <entry name="DayOfMonth"/>
                    <entry name="EndTime"/>
                    <entry name="StartTime"/>
                    <entry name="NextRun"/>
```

```
                    <entry name="RepeatesCount"/>
                </key>
              </key>
</key></value>
  </param>
</command>
.
```

5.5.2.1 *Jobs Scheduler Options*

Jobs Scheduler provides you with options to monitor and manage the scheduled jobs. Table 5-8 lists the BQL commands supported for Jobs Scheduler operations.

Table 5-8 Job Scheduler BQL Commands

Command Class	Action	Description
FindJobs	View	View and monitor the status of scheduled jobs.
RescheduleJob	Reschedule	Reschedule a job to a later date/time.
UpdateJob	Suspend—Calls the	Suspend a scheduled job.
	Resume—Calls the resumeJob	Resume a suspended job.
	Stop—Calls the cancelJob	Cancel a scheduled job.
DeleteJob	Delete	Delete a scheduled job.

5.5.3 Report Manager Interfaces

Table 5-9 lists the BQL commands supported for Report Manager queries.

Table 5-9 Supported Report Manager BQL Command Queries

Command Name	IMO/OID Type	IMO/OID Value	Description
Get	com.sheer.imo.IReportRoot	{[ReportRoot]}	Retrieves available reports (standard and user-defined).
Get	com.sheer.imo.IReportList	{[ReportList(ListTargetOid={[ReportRoot][ReportCategory(Category=*ReportCat egory*)]})]}	Retrieves generated reports for specific report category (Events Reports and Inventory Reports).

Command Name	IMO/OID Type	IMO/OID Value	Description
Get	com.sheer.imo.IReportList	{[ReportList(ListTargetOid={[ReportRoot][ReportCategory(Category=*ReportCategory*)][ReportType(Type=*R*	Retrieves generated reports for specific report type (Daily Event Count, Syslog Trends Report, and so on).

Table 5-10 lists the BQL commands supported for Report Manager operations.

Table 5-10 Supported Report Manager BQL Command Operations

Command Name	IMO/OID Type	IMO/OID Value	Description
AddReport Category	—	{[ReportRoot]}	Creates a new folder.
	—	{[ReportRoot][ReportCategory(Category=*ReportCategory*)]}	Creates a new report folder under an existing report folder (Events Reports, Inventory Reports, and user-defined folders).
SaveReportTypeCommand	—	{[ReportRoot][ReportCategory(Category=*ReportCategory*)]} Refer to the reports.xml file under *NETWORKHOME*/Main/registry for parameter details.	Creates a new user-defined report.
RunReportType Command	com.sheer.imo.IReport	— Refer to the reports.xml file under *NETWORKHOME*/Main/registry for parameter details.	Generates reports.
Rename	—	{[ReportRoot][ReportCategory(Category=*ReportCategory*)][ReportCategory(Category=*User-definedReportName*)]}	Renames the user-defined report folder name.
Move	—	{[ReportRoot][ReportCategory(Category=*ReportCategory*)][ReportData(Id=*ReportID*)]}	Moves the user-defined report under different report folder.

Command Name	IMO/OID Type	IMO/OID Value	Description
Delete	—	{[ReportList(ListTargetOid={[ReportRoot][ReportCategory(Category=*ReportCategory*)][ReportType(Type=*ReportType*))]})][Report(Id=*ReportID*)]}	Deletes generated (user-defined) reports.
Delete	—	{[ReportRoot][ReportCategory(Category=*ReportCategory*)][ReportCategory(Category=*User-defined Reports*)]}	Deletes user-defined report folder.

5.5.4 Samples BQL Scripts for Report

This section contains the following sample BQL scripts:

The Mediator Debugger tool helps you to identify the BQL commands for any Prime Network GUI task (except for Cisco Prime Network Events tasks). Using this tool, you can write your own BQL commands for the required GUI tasks.

Updating Report Manager Settings

The following example shows the usage of the BQL **UpdateReportSettings** command to update the purge (91 days and 29 MB) and security (true: to enable report sharing for other users) properties.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="UpdateReportSettings">
  <param name="minimalAge">
        <value>91</value>
  </param>
  <param name="maxDiskSpace">
        <value>29</value>
  </param>
  <param name="isReportsSecure">
        <value>true</value>
  </param>
  <param name="isDefaultValues">
        <value>false</value>
  </param>
</command>
```

.

Adding a New Report Folder Under a Standard Report Folder

The following example shows the usage of the BQL **AddReportCategory** command to add a new folder (User-defined Reports) under existing report folders (Inventory Reports).

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="AddReportCategory">
  <param name="categoryOid">
    <value>{[ReportRoot][ReportCategory(Category=Inventory
  Reports)]}</value>
  </param>
  <param name="categoryName">
        <value>User-defined Reports</value>
  </param>
</command>
```

.

Adding a New Report Folder

The following example shows the usage of the BQL **AddReportCategory** command to add a new folder (BQL-User-Defined) in the same level as standard report folders (ReportRoot).

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="AddReportCategory">
  <param name="categoryOid">
          <value>{[ReportRoot]}</value>
  </param>
  <param name="categoryName">
        <value>BQL-User-Defined</value>
  </param>
</command>
```

.

Creating a User-defined (Prime Network vs. Event Archive Statistics) Report

The following example shows the usage of the BQL **SaveeportTypeCommand** command to create a new report (BQL Sample Test - Prime Network vs Event) of the type decap_daily_statistics.rptdesign (Prime Netework vs. Event Archive Statistics) under the Events Reports category. See Specifying Date Format, page 158 to understand how to schedule.
Refer to the reports.xml file under *NETWORKHOME*/Main/registry for the *anaDesign* parameter.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="SaveReportTypeCommand">
  <param name="name">
        <value>BQL Sample Test - Prime Network vs Event</value>
  </param>
  <param name="description">
        <value>BQL Sample Test - Prime Network vs Event</value>
  </param>
  <param name="anaDesign">
        <value>decap_daily_statistics.rptdesign</value>
  </param>
  <param name="decapDesign">
        <value />
  </param>
  <param name="dataSource">
        <value>0</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="location">
        <value>{[ReportRoot][ReportCategory(Category=Events
      Reports)]}</value>
  </param>
  <param name="values">
  <value>
        <IReportParameter>
              <Name type="String">Date Group - OrderedDateRange</Name>
              <Value type="java.lang.String_Array">
                  <java.lang.String>3,1272695603000,1273646003000</java.la
            ng.String>
              </Value>
        </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">IsAllDates</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>false</java.lang.String>
        </Value>
  </IReportParameter>
</value>
<value />
  </param>
</command>
```

.

Creating a User-defined (Software Summary (by device)) Report Under User-defined Folder

The following example shows the usage of the BQL **SaveReportTypeCommand** command to create a new report (BQL-Software Summary by Device) of the type SoftwareVersionSummaryReport.rptdesign (Software Summary (by device)) under the user-defined folder (User-defined Reports), which is defined under standard report folder (Inventory Reports). This report is generated for ASR10k_SBC and C2900 network elements. See Specifying Date Format, page 158 to understand how to schedule.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="SaveReportTypeCommand">
  <param name="name">
        <value>BQL-Software Summary by Device</value>
  </param>
  <param name="description">
        <value>BQL-Software Summary by Device</value>
  </param>
  <param name="anaDesign">
        <value>SoftwareVersionSummaryReport.rptdesign</value>
  </param>
  <param name="decapDesign">
        <value />
  </param>
  <param name="dataSource">
        <value>2</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="location">
        <value>{[ReportRoot][ReportCategory(Category=Inventory
        Reports)][ReportCategory(Category=User-defined Reports)]}</value>
  </param>
  <param name="values">
        <value>
              <IReportParameter>
                  <Name type="String">SelectedDevices</Name>
                    <Value type="java.lang.String_Array">
                          <java.lang.String>ASR10k_SBC</java.lang.String>
                          <java.lang.String>C2900</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
  </param>
</command>
```

.

Generating Prime Network vs. Event Archive Statistics Report

The following example shows the usage of the BQL **RunReportTypeCommand** command to generate a Fault DB vs. Event Archive Statistics, page 148 report. See Specifying Date

Format, page 158 to understand how to schedule. Refer to the reports.xml file under NETWORKHOME/Main/registry for the type parameter.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="RunReportTypeCommand">
  <param name="name">
        <value>BQL- Prime Network vs Event Archive</value>
  </param>
  <param name="description">
        <value>BQL- Prime Network vs Event Archive</value>
  </param>
  <param name="type">
        <value>Prime Network vs. Event Archive Statistics</value>
  </param>
  <param name="source">
        <value>0</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="values">
        <value>
              <IReportParameter>
                    <Name type="String">ToDate</Name>
                    <Value type="java.lang.String_Array">
                        <java.lang.String>1273578242051</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
<value>
  <IReportParameter>
        <Name type="String">FromDate</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>1273146242051</java.lang.String>
        </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">IsAllDates</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>false</java.lang.String>
        </Value>
  </IReportParameter>
</value>
</param>
<param name="data">
  <value />
</param>
</command>
.
```

Generating Daily Event Count

The following example shows the usage of the BQL **RunReportTypeCommand** command to generate a Detailed Event Count (By Device), page 148 report. This report is generated

for ASR10k_SBC network element. See Specifying Date Format, page 158 to understand the schedule date format.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="RunReportTypeCommand">
  <param name="name">
        <value>BQL Daily Event Count</value>
  </param>
  <param name="description">
        <value>BQL Daily Event Count</value>
  </param>
  <param name="type">
        <value>Daily Event Count</value>
  </param>
  <param name="source">
        <value>0</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="values">
        <value>
              <IReportParameter>
                    <Name type="String">ToDate</Name>
                    <Value type="java.lang.String_Array">
                        <java.lang.String>1273578602509</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
<value>
  <IReportParameter>
        <Name type="String">FromDate</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>1273146602509</java.lang.String>
        </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">IsAllDates</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>false</java.lang.String>
        </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">SelectedDevices</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>'{[ManagedElement(Key=ASR10k_SBC)]}'</jav
        a.lang.String>
        </Value>
  </IReportParameter>
</value>
</param>
<param name="data">
  <value />
```

```
</param>
</command>
.
```

Generating Detailed Traps Report

The following example shows the usage of the BQL **RunReportTypeCommand** command to generate a Detailed Traps, page 149 report. This report is generated on event archive (source = 1) for ASR10k_SBC and C2900 network elements. See Specifying Date Format, page 158 and Detailed Traps, page 149 report to understand how to schedule and specify the trap versions respectively. Refer to the reports.xml file under *NETWORKHOME*/Main/registry for the *type* parameter.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="RunReportTypeCommand">
  <param name="name">
        <value>BQL Test Trap</value>
  </param>
  <param name="description">
        <value>BQL Test Trap</value>
  </param>
  <param name="type">
        <value>Detailed Traps</value>
  </param>
  <param name="source">
        <value>1</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="values">
        <value>
              <IReportParameter>
                    <Name type="String">ToDate</Name>
                    <Value type="java.lang.String_Array">
                        <java.lang.String>1273585760756</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
<value>
  <IReportParameter>
        <Name type="String">FromDate</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>1273326560756</java.lang.String>
        </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">IsAllDates</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>false</java.lang.String>
        </Value>
  </IReportParameter>
</value>
```

```
<value>
  <IReportParameter>
        <Name type="String">SelectedDevices</Name>
        <Value type="java.lang.String_Array">
<java.lang.String>'{[ManagedElement(Key=ASR10k_SBC)]}'</java.lang.String>
        <java.lang.String>'{[ManagedElement(Key=C2900)]}'</java.lang.Stri
      ng>
        </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">version</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>'1'</java.lang.String>
              <java.lang.String>'2'</java.lang.String>
              <java.lang.String>'3'</java.lang.String>
        </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">vgeneric</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>'0'</java.lang.String>
              <java.lang.String>'1'</java.lang.String>
              <java.lang.String>'2'</java.lang.String>
              <java.lang.String>'3'</java.lang.String>
              <java.lang.String>'4'</java.lang.String>
              <java.lang.String>'5'</java.lang.String>
              <java.lang.String>'6'</java.lang.String>
        </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">VSpecific</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>1,3,6,1,6,3,1,1,6</java.lang.String>
        </Value>
  </IReportParameter>
</value>
</param>
  <param name="data">
        <value />
  </param>
</command>
.
```

Generating Devices with the Most Syslogs

The following example shows the usage of the BQL **RunReportTypeCommand** command to generate Devices with the Most Syslogs, page 150 report. You can specify source for the syslog data. It can be from either Alarm DB (source = 0) or Event Archive (source = 1).

This report is generated for ASR10k_SBC and C2900 network elements. See Specifying Date Format, page 158 to understand how to schedule.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="RunReportTypeCommand">
  <param name="name">
        <value>BQL - Most Syslog</value>
  </param>
  <param name="description">
        <value>BQL - Most Syslog</value>
  </param>
  <param name="type">
        <value>Devices with the Most Syslogs</value>
  </param>
  <param name="source">
        <value>1</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="values">
        <value>
              <IReportParameter>
                    <Name type="String">ToDate</Name>
                    <Value type="java.lang.String_Array">
                        <java.lang.String>1273586526196</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
<value>
  <IReportParameter>
        <Name type="String">FromDate</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>1273240926196</java.lang.String>
        </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">IsAllDates</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>false</java.lang.String>
        </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">numberOfDevices</Name>
        <Value type="java.lang.String_Array">
              <java.lang.String>2</java.lang.String>
        </Value>
  </IReportParameter>
</value>
<value>
  <IReportParameter>
        <Name type="String">SelectedDevices</Name>
        <Value type="java.lang.String_Array">
```

```
            <java.lang.String>'{[ManagedElement(Key=ASR10k_SBC)]}'</jav
            a.lang.String>
            <java.lang.String>'{[ManagedElement(Key=C2900)]}'</java.lan
            g.String>
        </Value>
  </IReportParameter>
</value>
</param>
  <param name="data">
        <value />
  </param>
</command>
.
```

Generating Most Commonly Daily Events

The following example shows the usage of the BQL **RunReportTypeCommand** command to generate a Most Common Daily Events, page 151 report. You can specify source for the events data. It can be from either Alarm DB (source = 0) or Event Archive (source = 1). You can also display the report data in pie chart format. This report is generated for ASR10k_SBC and C2900 network elements. See Specifying Date Format, page 158 to understand how to schedule.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="RunReportTypeCommand">
  <param name="name">
        <value>BQL Test - Daily Events</value>
  </param>
  <param name="description">
        <value>BQL Test - Daily Events</value>
  </param>
  <param name="type">
        <value>Most Common Daily Events</value>
  </param>
  <param name="source">
        <value>0</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="values">
        <value>
              <IReportParameter>
                    <Name type="String">ToDate</Name>
                    <Value type="java.lang.String_Array">
                        <java.lang.String>1273587255017</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
        <value>
              <IReportParameter>
                    <Name type="String">FromDate</Name>
                    <Value type="java.lang.String_Array">
                        <java.lang.String>1273414455017</java.lang.String>
                    </Value>
```

```
            </IReportParameter>
        </value>
        <value>
            <IReportParameter>
                <Name type="String">IsAllDates</Name>
                <Value type="java.lang.String_Array">
                    <java.lang.String>false</java.lang.String>
                </Value>
            </IReportParameter>
        </value>
        <value>
            <IReportParameter>
                <Name type="String">numberOfMessages</Name>
                <Value type="java.lang.String_Array">
                    <java.lang.String>2</java.lang.String>
                </Value>
            </IReportParameter>
        </value>
        <value>
            <IReportParameter>
                <Name type="String">Show Charts</Name>
                <Value type="java.lang.String_Array">
                    <java.lang.String>true</java.lang.String>
                </Value>
            </IReportParameter>
        </value>
  </param>
  <param name="data">
        <value />
  </param>
</command>
  .
```

Generating Syslog Trends Report

The following example shows the usage of the BQL **RunReportTypeCommand** command to generate a Syslog Trend (By Severity), page 152 report. You can specify source for the events data. It can be from either Alarm DB (source = 0) or Event Archive (source = 1). This report is generated for ASR10k_SBC and C2900 network elements. See Specifying Date Format, page 158 and Syslog Trend (By Severity), page 152 to understand how to schedule and specify syslog severity respectively.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="RunReportTypeCommand">
  <param name="name">
        <value>BQL Test - Syslog Trend</value>
  </param>
  <param name="description">
        <value />
  </param>
  <param name="type">
        <value>Syslog Trend (by severity)</value>
  </param>
  <param name="source">
        <value>0</value>
```

```
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="values">
        <value>
              <IReportParameter>
                    <Name type="String">ToDate</Name>
                    <Value type="java.lang.String_Array">
                        <java.lang.String>1273587731211</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
        <value>
              <IReportParameter>
                    <Name type="String">FromDate</Name>
                    <Value type="java.lang.String_Array">
                      <java.lang.String>1273501331211</java.lang.String>
                    </Value>
              </IReportParameter>
         </value>
         <value>
              <IReportParameter>
                    <Name type="String">IsAllDates</Name>
                    <Value type="java.lang.String_Array">
                          <java.lang.String>false</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
        <value>
              <IReportParameter>
                    <Name type="String">SelectedDevices</Name>
                    <Value type="java.lang.String_Array">
<java.lang.String>'{[ManagedElement(Key=ASR10k_SBC)]}'</java.lang.String>
                          <java.lang.String>'{[ManagedElement(Key=C2900)]
                    }'</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
        <value>
              <IReportParameter>
                    <Name type="String">IntervalUnit</Name>
                    <Value type="java.lang.String_Array">
                          <java.lang.String>Days</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
        <value>
              <IReportParameter>
                    <Name type="String">SelectedSeverity</Name>
                    <Value type="java.lang.String_Array">
                          <java.lang.String>'4'</java.lang.String>
                          <java.lang.String>'5'</java.lang.String>
                          <java.lang.String>'6'</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
```

```
        <value>
              <IReportParameter>
                    <Name type="String">SelectedSyslogMsgs</Name>
                    <Value type="java.lang.String_Array">
                          <java.lang.String>'A generic alert-level message
                    syslog'</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
  </param>
  <param name="data">
        <value />
  </param>
</command>
.
```

Generating Software Summary (by version) Report

The following example shows the usage of the BQL **RunReportTypeCommand** command to generate a Software Summary (By Version), page 154 report. This report is generated for ASR10k_SBC and C2900 network elements.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="RunReportTypeCommand">
  <param name="name">
        <value>BQL Software Version</value>
  </param>
  <param name="description">
        <value>BQL Software Version</value>
  </param>
  <param name="type">
        <value>Software Summary (by version)</value>
  </param>
  <param name="source">
        <value>2</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="values">
        <value>
              <IReportParameter>
                    <Name type="String">SelectedDevices</Name>
                    <Value type="java.lang.String_Array">
                          <java.lang.String>'{[ManagedElement(Key=ASR10k_S
                          BC)]}'</java.lang.String>
                          <java.lang.String>'{[ManagedElement(Key=C2900)]
                          }'</java.lang.String>
                    </Value>
              </IReportParameter>
         </value>
  </param>
  <param name="data">
        <value />
  </param>
</command>
```

.

Generating Reports for Unmanaged Devices

The following example shows the usage of the BQL **RunReportTypeCommand** command to generate a Software Summary (By Version), page 154 report. This report is generated for ASR10k_SBC and C2900 network elements.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="RunReportTypeCommand">
  <param name="name">
        <value>My Detailed Syslogs</value>
  </param>
  <param name="description">
        <value />
  </param>
  <param name="type">
        <value>Detailed Syslogs</value>
  </param>
  <param name="source">
        <value>1</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="values">
        <value>
              <IReportParameter type="IReportParameter" instance_id="0">
                    <Name type="String">ToDate</Name>
                    <Value type="java.lang.String_Array">
                        <java.lang.String>1294825776419</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
        <value>
              <IReportParameter type="IReportParameter" instance_id="0">
                    <Name type="String">FromDate</Name>
                    <Value type="java.lang.String_Array">
                        <java.lang.String>1294739376419</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
        <value>
              <IReportParameter type="IReportParameter" instance_id="0">
                    <Name type="String">SelectedDevices</Name>
                    <Value type="java.lang.String_Array">
                          <java.lang.String>'{[ManagedElement(Key=c7-
                          sw1)]}'</java.lang.String>
                          <java.lang.String>'{[ManagedElement(Key=c7-
                          sw3)]}'</java.lang.String>
                          <java.lang.String>'{[ManagedElement(Key=c7-
                          sw4)]}'</java.lang.String>
<java.lang.String>'{[ManagedElement(Key=10.56.59.140)]}'</java.lang.String
>
<java.lang.String>'{[ManagedElement(Key=10.56.59.52)]}'</java.lang.String>

<java.lang.String>'{[UnmanagedElement(IP=11.1.1.1)]}'</java.lang.String>
```

```
<java.lang.String>'{[UnmanagedElement(IP=22.2.2.2)]}'</java.lang.String>

<java.lang.String>'{[UnmanagedElement(IP=33.3.3.3)]}'</java.lang.String>
                    </Value>
              </IReportParameter>
        </value>
        <value>
              <IReportParameter type="IReportParameter" instance_id="0">
                    <Name type="String">rawData</Name>
                    <Value type="java.lang.String_Array">
                          <java.lang.String />
                    </Value>
              </IReportParameter>
        </value>
        <value />
  </param>
  <param name="data">
        <value />
  </param>
</command>
```

Creating VPLS Detailed Report

The following example shows the usage of the BQL **SaveReportTypeCommand** command to create a new report (VPLS Detailed Report) of the type detailed-vpls-report.rptdesign (VPLS Detailed Report) under the standard report folder (Network Service Reports). This report is generated on a map *maptest* and for the VPLS name *bgp*.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="SaveReportTypeCommand">
  <param name="location">
    <value>{[ReportRoot][ReportCategory(Category=Network Service
    Reports)]}</value>
  </param>
  <param name="name">
        <value>VPLS Detailed Report</value>
  </param>
  <param name="description">
        <value />
  </param>
  <param name="anaDesign">
        <value>detailed-vpls-report.rptdesign</value>
  </param>
  <param name="decapDesign">
        <value />
  </param>
  <param name="dataSource">
        <value>0</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="values">
    <value>
        <IReportParameter>
```

```
                <Name type="String">vplsName</Name>
                <Value type="java.lang.String_Array">
                      <java.lang.String>bgp</java.lang.String>
                </Value>
          </IReportParameter>
        </value>
      <value>
          <IReportParameter>
                <Name type="String">BusinessTag</Name>
                <Value type="java.lang.String_Array">
                      <java.lang.String />
                </Value>
          </IReportParameter>
        </value>
      <value>
          <IReportParameter>
                <Name type="String">MapList</Name>
                <Value type="java.lang.String_Array">
                      <java.lang.String>'maptest'</java.lang.String>
                </Value>
          </IReportParameter>
        </value>
  </param>
</command>
```

.

Getting ATOM Summary Report without Filters

The following example shows the usage of the BQL **RunReportTypeCommand** command to generate an AToM Summary, page 155 report without enabling the filter.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="RunReportTypeCommand">
  <param name="name">
        <value>BQL ATOM Summary</value>
  </param>
  <param name="description">
        <value />
  </param>
  <param name="type">
        <value>AToM Summary</value>
  </param>
  <param name="source">
        <value>0</value>
  </param>
  <param name="public">
        <value>false</value>
  </param>
  <param name="values">
        <value>
              <IReportParameter>
                    <Name type="String">PseudowireType</Name>
                    <Value type="java.lang.String_Array">
                          <java.lang.String />
                    </Value>
              </IReportParameter>
        </value>
        <value>
```

```
            <IReportParameter>
                  <Name type="String">BusinessTag</Name>
                  <Value type="java.lang.String_Array">
                        <java.lang.String />
                  </Value>
            </IReportParameter>
      </value>
      <value>
            <IReportParameter>
                  <Name type="String">PseudowireName</Name>
                  <Value type="java.lang.String_Array">
                        <java.lang.String />
                  </Value>
            </IReportParameter>
      </value>
      <value>
            <IReportParameter>
                  <Name type="String">MapList</Name>
                  <Value type="java.lang.String_Array" />
            </IReportParameter>
      </value>
  </param>
  <param name="data">
        <value />
  </param>
</command>
.
```

Listing Reports

The following example shows the usage of the BQL **Get** command to retrieve all report categories and report types.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Get">
  <param name="oid">
        <value>{[ReportRoot]}</value>
  </param>
  <param name="rs">
        <value><key name="">
      <entry name="depth">0</entry>
      <entry name="register">true</entry>
      <entry name="cachedResultAcceptable">false</entry>
      <key name="requiredProperties">
        <key name="*">
              <entry name="*"/>
        </key>
      </key>
      </key></value>
  </param>
</command>
.
```

Listing a Specific Report Category (Events Reports)

The following example shows the usage of the BQL **Get** command to retrieve all generated events reports.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Get">
  <param name="oid">
    <value>{[ReportList(ListTargetOid={[ReportRoot][ReportCategory(Catego
  ry=EventsReports)]})]}</value>
  </param>
  <param name="rs">
        <value><key name="">
      <entry name="depth">0</entry>
      <entry name="register">true</entry>
      <entry name="cachedResultAcceptable">false</entry>
      <key name="requiredProperties">
        <key name="com.sheer.imo.IReport">
              <entry name="*"/>
        </key>
        <key name="com.sheer.imo.IReportList">
              <entry name="*"/>
        </key>
      </key>
      <key name="excludedProperties">
        <key name="com.sheer.imo.IReport">
              <entry name="URI"/>
        </key>
      </key>
  </key></value>
        </param>
</command>
.
```

Note If you get an empty list as the output, it means that reports were not generated.

Listing a Specific Report Type (Prime Network vs. Event Archive Statistics)

The following example shows the usage of the BQL **Get** command to retrieve all generated Prime Network vs. Event Archive Statistics reports.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Get">
  <param name="oid">
    <value>{[ReportList(ListTargetOid={[ReportRoot][ReportCategory(Catego
    ry=Events Reports)][ReportType(Type=Prime Network vs. Event Archive
    Statistics)]})]}</value>
  </param>
  <param name="rs">
        <value><key name="">
  <entry name="depth">0</entry>
  <entry name="register">true</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
        <key name="com.sheer.imo.IReport">
              <entry name="*"/>
        </key>
        <key name="com.sheer.imo.IReportList">
```

```
              <entry name="*"/>
        </key>
  </key>
  <key name="excludedProperties">
        <key name="com.sheer.imo.IReport">
              <entry name="URI"/>
        </key>
  </key>
</key></value>
  </param>
</command>
.
```

Renaming User-defined Folder

The following example shows the usage of the BQL **Rename** command to rename a user-defined report folder.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Rename">
  <param name="targetOid">
        <value>{[ReportRoot][ReportCategory(Category=InventoryReports)][R
      eportCategory(Category=BQL-Reports)]}</value>
  </param>
  <param name="newName">
        <value>BQL-Reports-Sample</value>
  </param>
</command>
.Moving User-defined Report
```

The following example shows the usage of the BQL **Move** command to move a user-defined report.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Move">
  <param name="selectedOids">
        <value>{[ReportRoot][ReportCategory(Category=EventsReports)][Repor
      tData(Id=5)]}</value>
  </param>
  <param name="destinationOid">
        <value>{[ReportRoot][ReportCategory(Category=Inventory
      Reports)]}</value>
  </param>
</command>
.
```

Deleting User-defined Reports

The following example shows the usage of the BQL **Delete** command to delete generated (user-defined) reports.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Delete">
  <param name="oids">
```

```
    <value>{[ReportList(ListTargetOid={[ReportRoot][ReportCategory(Categor
    y=Events Reports)][ReportType(Type=Prime Network vs. Event Archive
    Statistics)]})][Report(Id=9)]}</value>
    <value>{[ReportList(ListTargetOid={[ReportRoot][ReportCategory(Categor
    y=Events Reports)][ReportType(Type=Prime Network vs. Event Archive
    Statistics)]})][Report(Id=8)]}</value>
    <value>{[ReportList(ListTargetOid={[ReportRoot][ReportCategory(Categor
    y=Events Reports)][ReportType(Type=Prime Network vs. Event Archive
    Statistics)]})][Report(Id=6)]}</value>
  </param>
</command>
.
```

Deleting User-defined Report Folder

The following example shows the usage of the BQL **Delete** command to delete a user-defined report folder.

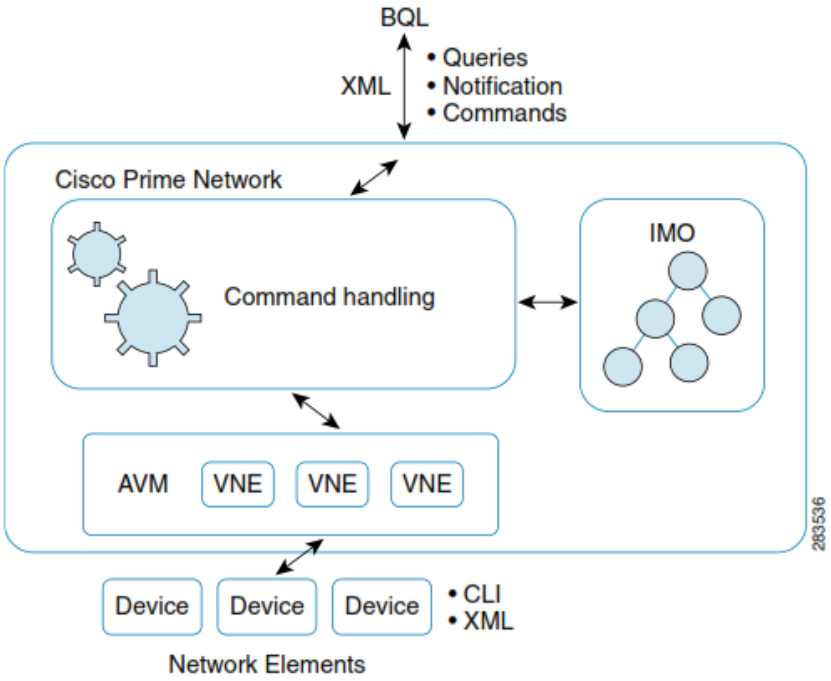
```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Delete">
  <param name="oids">
        <value>{[ReportRoot][ReportCategory(Category=InventoryReports)][Re
      portCategory(Category=User-defined Reports)]}</value>
  </param>
</command>
.
```

5.6 Running Command Builder Scripts Using BQL

Cisco Prime Network packages a set of default commands (scripts), which can be used to configure and troubleshoot your devices. Commands enable you to execute a programmable sequence of SNMP or Telnet command lines. These commands can include data properties taken from the Prime Network information model (built-in), as well as user-defined input parameters entered during runtime.
Commands can be associated with any existing object group (IMO), type, or instance. This associated item is the working object on which the command is developed and tested. It enables access to the live information of the network object with which the command is associated.

Figure 5-8 shows how you can handle commands using BQL.

Figure 5-8 Handling Commands using BQL



Sometimes there is a need to add user-defined commands, for example to invoke non default commands for automation of activation or configuration services. Command Builder is the tool to create these new commands. You can invoke the Command Builder tool and run the commands using the BQL commands.

Command Builder is designed to enable you to create new commands. It utilizes regular Prime Network interfaces to the network. The commands that you define can be used to make multiple configuration changes on a single network element.

For example, if you want to set the duplex mode on an interface, you can create a command that has the network element-specific CLI and script-based implementation to create the setDuplex() operation.

Command definitions can be overloaded; that is, you can create a command for setDuplex() specific to Cisco 7600 router, and you can also create a command for setDuplex() on a Juniper router. The actual implementations are different, but they are each registered in the system to be mapped to the appropriate network element. On GUIs or integration interfaces, you see setDuplex() as a single action that you can run on one of the network elements. When setDuplex() is called, the call is routed to the VNE layer, which looks up the implementation of setDuplex() for the specific network element, for example a Cisco 7600, and the VNE calls setDuplex_7600().

Additional Reading

- Review *Cisco Prime Network 4.2.2 Administrator Guide* to understand the Prime Network user roles and scopes.
- Review *Cisco Prime Network 4.2.2 Customization Guide* to understand the Command Builder application.
- See the *Cisco Prime Network Information Model* Javadoc to understand the IMO for VNE and Command Builder commands (scripts). This document is available on the Prime Network Technology Center website. You must have a Cisco.com account with partner level access, or you must be a Prime Network licensee to access this website.

5.6.1 Command Builder Interfaces

Table 5-11 lists the BQL commands supported for Command Builder queries.

Table 5-11 Supported Command Builder BQL Command Queries

Command Name	IMO/OID Type	IMO/OID Value	Description
Get	com.sheer.imo.keys.IScriptOid	{[Script(Name=*script-name*)(RegistryPath=*AVM*/agents/da/*DeviceName*/imo/scripts/com.sheer.imo.IManagedElement)]}	Retrieves commands.
GetPublishedElementVersions	com.sheer.imo.keys.IScriptOid	{[Script(Name=*script-name*)(RegistryPath=*AVM*/agents/da/*DeviceName*/imo/scripts/com.sheer.imo.IManagedElement)]}	Retrieves versions of published commands.
GetAvailableScriptParameters	com.sheer.imo.keys.IManagedElementOid	{[ManagedElement(Key=*DeviceName*)]}	Retrieves available parameters for a specified NE.

Table 5-12 lists the BQL commands supported for Command Builder operations.

Table 5-12 Supported Command Builder BQL Command Operations

Command Name	IMO/OID Type	IMO/OID Value	Description
Set	com.sheer.imo.keys.IScriptOid	{[Script(ContextImoType=com.sheer.imo.IManagedElement)(Name=*script-name*)]}	Creates or edits a command.
Delete	com.sheer.imo.keys.IScriptOid	{[Script(Name=*script-name*)(RegistryPath=*AVM*/agents/da/*DeviceName*/imo/scripts/com.sheer.imo.IManagedElement)]}	Deletes a command.
Command name	com.sheer.imo.keys.IManagedElementOid, [Parameters,….]	{[ManagedElement(Key=*DeviceName*)]}, [Parameter Values,….]	Runs a command.

5.6.2 Credentials used for Command Builder Scripts Executions

By default, when connecting to devices in order to run command scripts, Cisco Prime Network uses the credentials specified for the VNE.
Cisco Prime Network allows to override this default setting, and use the executing user's credentials.

- In order to use the executing user's credentials, add 2 parameters named DEVICE_USER_NAME and DEVICE_PASSWORD to the BQL

Example:

```
<command name="show-ver">
    <param name="oid">
        <value>{[ManagedElement(Key=10.56.101.75)]}</value>
    </param>
    <param name="DEVICE_PASSWORD">
        <value>general</value>
    </param>
    <param name="DEVICE_USER_NAME">
        <value>general</value>
    </param>
</command>
```

Note
User credentials are not supported for command builder scripts written in SNMP syntax.

5.6.3 Sample BQL Scripts for Command Builder Commands

This section contains the following sample BQL scripts:

- Deleting a Command, page 192

The Mediator Debugger tool helps you identify the BQL commands for any Prime Network GUI task (except for Cisco Prime Network Events tasks). Using this tool, you can write your own BQL commands for the required GUI tasks.

Creating a Command

The following example shows the usage of the BQL **Set** command to create a command builder script (ShowVRF). In this example:

- Script is used to run the command `show ip vrf $`*vrfname*`$` on the managed element ASR10k_SBC; where *vrfname* is the mandatory parameter.
- Language used is Cisco Prime Network Macro (0) and protocol is Telnet (0).
- Command authorization is assigned to Administrator.
- Timeout is 180000 milliseconds (3 minutes).

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Set">
  <param name="imo">
        <value>
        <IScript>
              <IDtype="Oid">{[Script(ContextImoType=com.sheer.imo.IManage
              dElement)(Name=ShowVRF)]}</ID>
              <ActivationScript type="String">show ip vrf
              $vrfname$</ActivationScript>
              <ErrorCondition type="String" />
              <Language type="Integer">0</Language>
              <MenuCaption type="String">Show VRF</MenuCaption>
              <MenuPath type="String">VRF Commands</MenuPath>
              <MenuVisible type="Boolean">true</MenuVisible>
              <Parameters type="IMObjects_Array">
                    <IScriptParameter>
                          <ID type="Oid">{[ScriptParameter(Index=0)]}</ID>
                          <Caption type="String">VRF Name</Caption>
                          <DefaultValue type="String" />
                          <DisplayWidth type="Integer">15</DisplayWidth>
                          <EnumValues type="">Null</EnumValues>
                          <Name type="String">vrfname</Name>
                          <OnPopulateScript
                          type="">Null</OnPopulateScript>
                          <OnPopulateScriptFileName
                          type="">Null</OnPopulateScriptFileName>
                          <OnValidateScript
                          type="">Null</OnValidateScript>
                          <OnValidateScriptFileName
                          type="">Null</OnValidateScriptFileName>
                          <Page type="String">vrfname</Page>
                          <Required type="Boolean">true</Required>
                          <Tooltip type="String">Enter VRF Name</Tooltip>
                          <Type type="String">java.lang.String</Type>
                          <Visible type="Boolean">true</Visible>
                    </IScriptParameter>
              </Parameters>
              <Protocol type="Integer">0</Protocol>
```

```
              <Roles type="java.lang.String_Array">
                  <java.lang.String>Administrator</java.lang.String>
              </Roles>
              <RollbackScript type="String" />
              <TabPages type="IMObjects_Array">
                    <IScriptTabPage>
                        <ID
                        type="Oid">{[ScriptTabPage(Name=vrfname)]}</ID>
                        <PageName type="String">vrfname</PageName>
                        <Parameters type="java.lang.String_Array">
                          <java.lang.String>rd</java.lang.String>
                          <java.lang.String>rt</java.lang.String>
                          <java.lang.String>vrfname</java.lang.String>
                          </Parameters>
                    </IScriptTabPage>
              </TabPages>
              <Timeout type="Integer">180000</Timeout>
        </IScript>
      </value>
  </param>
  <param name="neOid">
        <value>{[ManagedElement(Key=ASR10k_SBC)]}</value>
  </param>
  <param name="replace">
        <value>true</value>
  </param>
</command>
.
```

Running a Command

The following example shows how to run a command builder script (ShowVRF) with **aaa** as the input value for the *vrfname* parameter.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="ShowVRF">
  <param name="oid">
        <value>{[ManagedElement(Key=ASR10k_SBC)]}</value>
  </param>
  <param name="vrfname">
        <value>aaa</value>
  </param>
</command>
.
```

Getting a Command Parameter

The following example shows the usage of the BQL **GetAvailableScriptParameters** command to retrieve the available command builder script parameters for a specified NE.

```
<command name="GetAvailableScriptParameters">
  <param name="oid">
        <value>{[ManagedElement(Key=ASR10k_SBC)]}</value>
  </param>
</command>
.
```

Getting a Command

The following example shows the usage of the BQL **Get** command to retrieve a command builder script details for a specified NE.

```
<command name="Get">
  <param name="oid">
    <value>{[Script(ContextImoType=com.sheer.imo.IManagedElement)(Name=Sho
    wVRF)(RegistryPath=avm111/agents/da/ASR10k_SBC/imo/scripts)]}</value>
  </param>
  <param name="rs">
        <value><key name="">
  <entry name="depth">0</entry>
  <entry name="register">false</entry>
  <entry name="cachedResultAcceptable">false</entry>
  <key name="requiredProperties">
        <key name="*">
              <entry name="*"/>
        </key>
  </key>
</key></value>
  </param>
</command>
.
```

Deleting a Command

The following example shows the usage of the BQL **Delete** command to delete a command builder script for a specified NE.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Delete">
  <param name="oid">
<value>{[Script(Name=ShowVRF)(RegistryPath=avm111/agents/da/ASR10k_SBC/imo/
scripts/com.she er.imo.IManagedElement)]}</value>
  </param>
</command>
.
```

5.6.4 Command Builder Scripts Session Control

5.6.4.1 Additions to version 4.2.2

Cisco Prime Netowork allows you to control the script execution. Following statuses are added in Prime Network 4.2.2:

IScriptResult.getStatusEnum	Stands for
5	"Added to Queue"
6	"Aborted due to User Request"
7	"Execution Started"
8	"Aborted due to Timeout"

The “IScriptResult.getPlaceInQueue” script is also added in the Prime Network 4.2.2 version. This script holds the place of the script in the script execution queue. 0 is the place that is being executed. 1 is the next script that will be executed.

New options are added to abort a script that is still in the queue and to set a timeout for the execution of a script. After the timeout, if the script is still in the queue, it will not be executed.

5.6.4.2 Receiving session control notifications

When running a command builder script, you can choose to run it with notificationMode parameter set to true. For example:

```
<command name="show-ip-ospf-ne">
    <param name="oid">
        <value>{[ManagedElement(Key=10.56.23.48)]}</value>
    </param>
    <param name="notificationMode">
        <value>true</value>
    </param>
</command>
```

A new IMO table is added with the scripts execution status for a VNE. You can run the get command and register on the VNE and receive session control notification for all scripts that are executing on the VNE:

```
<command name="Get">
<param
name="oid"><value>{[ManagedElement(Key=10.56.23.48)]}</value></param>
<param name="rs"><value>
<key name="session-control-vne">
<entry name="depth">10000</entry>
<entry name="register">true</entry>
<entry name="cachedResultAcceptable">false</entry>
      <key name="requiredProperties">
            <key name="com.sheer.imo.IManagedElement">
            <entry name="ScriptResult"/>
            </key>
            <key name="com.sheer.imo.IScriptResult">
            <entry name="*"/>
            </key>
            <key name="com.sheer.imo.IScriptResultContainerAspect">
            <entry name="*"/>
            </key>
            <key name="com.sheer.imo.IScriptResultContainerAspectOid">
            <entry name="*"/>
            </key>
      </key>
</key></value></param>
</command>
```

.

By default, the ScriptResultContainer erases all scripts, which are in the queue for more than 5 minutes.

To change the interval for the whole set up, run the reg-tool command:

./runRegTool.sh 0.0.0.0 set site/agentdefaults/da/ScriptExecutionResultContainer/cleanupInterval <timeInMillis>

5.6.4.3 How can I set a timeout for the execution of a script?

You can set a time limit for a script that is in the queue, so that after the script is timed-out, it will not be executed.
For example, you run the following script:

```
<command name="show-ip-ospf-ne">
    <param name="oid">
        <value>{[ManagedElement(Key=10.56.23.48)]}</value>
    </param>
    <param name="timeout">
        <value>123456789</value>
    </param>
</command>
```

Timeout value is in miliseconds. It can be any integer up to 2^{63}-1.
If the script is still in queue after it is timed-out,the script will not be executed.
Timeout per script execution as seen here can be used to determine whether a timeout was specified in the definition of the command builder script.
If timeout is not specified per execution, the timeout that is defined during the creation of the script will be used. If timeout was not defined during the script execution or when defining the script, the script will never abort.

5.6.4.4 Examples

Example for “execution started notification” status enum=7 notification:

```
<IMObjects_Array>
  <IScalarNotification type="IScalarNotification" instance_id="0">
    <ID
type="Oid">{[Notification(SequenceNumber=1881)(Time=1391073381666)]}</I
D>
    <NewIMO type="IScriptResult" instance_id="1">
      <ID
type="Oid">{[ManagedElement(Key=10.56.23.48)][ScriptResultContainerAspe
ct][ScriptResult(ScriptName=show-ip-ospf-ne)(Sequence=4)]}</ID>
      <StatusEnum type="Integer">7</StatusEnum>
    </NewIMO>
    <PropertyName type="String">StatusEnum</PropertyName>
  </IScalarNotification>
</IMObjects_Array>
```

5.6.4.5 How can I abort a script?

Only scripts that are in queue can be aborted. You need the script OID and the right permissions to abort a script that is in queue. The script OID is available in the notifications sent when the script was added to queue. The script OID from the previous section is:

```
{[ManagedElement(Key=10.56.23.48)][ScriptResultContainerAspect][ScriptR
esult(ScriptName=show-ip-ospf-ne)(Sequence=4)]}
```

Only a script executer and an administrator can abort a script.

5.6.4.5.1 Aborting a script as the script executer

Use the following command to abort the script that you have executed:

```
<command name="UserAbortScript">
    <param name="oid">
<value>{[ManagedElement(Key=10.56.23.48)][ScriptResultContainerAspect][
ScriptResult(ScriptName=show-ip-ospf-ne)(Sequence=4)]}</value>
    </param>
</command>
```

5.6.4.5.2 Aborting a script as the administrator

The users with administrator role in Cisco Prime Network can abort any script. Using the script OID, run the following command:

```
<command name="AdminAbortScript">
    <param name="oid">
<value>{[ManagedElement(Key=10.56.23.48)][ScriptResultContainerAspect][
ScriptResult(ScriptName=show-ip-ospf-ne)(Sequence=4)]}</value>
    </param>
</command>
```

5.6.4.5.3 Possible responses for the aborting commands.

Following are the responses that are received for the aborting commands:

5.6.4.5.3.1 Response 1: failed admin abort.

```
<IGenericImo type="IGenericImo" instance_id="0">
 <ID type="Oid">{[GenericImo(Id=0)]}</ID>
 <StringValue type="String">Failed to abort script
{[ManagedElement(Key=event)][ScriptResultContainerAspect][ScriptResult(ScriptName=
!Device_ShowUsers)(Sequence=1375128523290)]}. Only scripts in Added_to_Queue
status could be aborted. </StringValue>
</IGenericImo>
```

5.6.4.5.3.2 Response 2: failed user abort

```
<IGenericImo type="IGenericImo" instance_id="0">
 <ID type="Oid">{[GenericImo(Id=0)]}</ID>
 <StringValue type="String">Failed to abort script
{[ManagedElement(Key=event)][ScriptResultContainerAspect][ScriptResult(ScriptName=
!Device_ShowUsers)(Sequence=1375128523290)]}. Only scripts in Added_to_Queue
status could be aborted. Only the user who operated the script could abort
it</StringValue>
</IGenericImo>
```

5.6.4.5.3.3 Response 3: abort succeeded

```
<IGenericImo type="IGenericImo" instance_id="0">
```

```
 <ID type="Oid">{[GenericImo(Id=0)]}</ID>
 <StringValue type="String">Successfuly aborted
script{[ManagedElement(Key=event)][ScriptResultContainerAspect][ScriptResult(ScriptN
ame=!Device_ShowUsers)(Sequence=1375128523290)]} </StringValue>
</IGenericImo>
```

5.6.4.5.3.4 Response 4: failed admin and user abort

```
<IGenericImo type="IGenericImo" instance_id="0">
 <ID type="Oid">{[GenericImo(Id=0)]}</ID>
 <StringValue type="String">No such script found on this managed
element</StringValue>
</IGenericImo>
```

5.7 Managing Soft Properties Using BQL

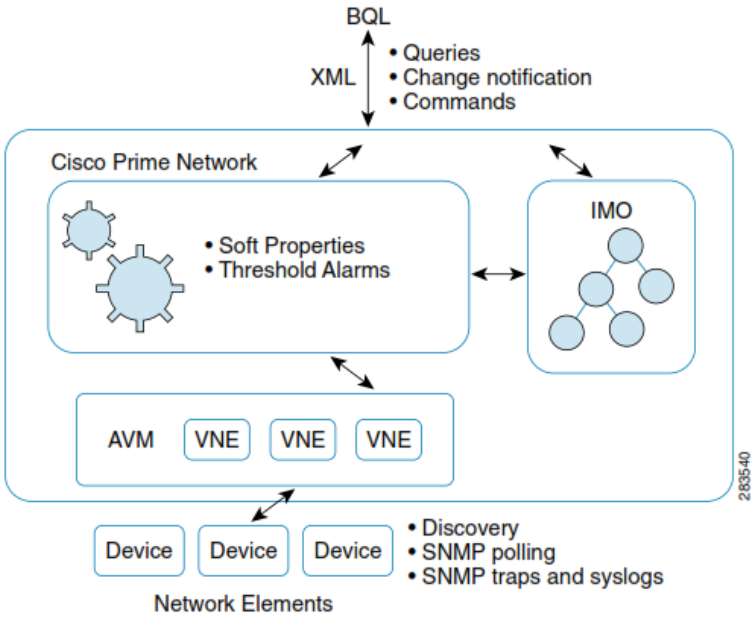
An NE is modeled as an interconnected hierarchy of Device Components (DCs), both physical (for example, cards and ports) and logical (for example, forwarding tables and profiles). Each DC maintains a set of properties, which contain its actual data (such as status, configuration, or performance).

You can extend the NE data collection and modeling by adding new properties to the DCs, and assigning them to NE MIB variables in runtime. The new soft properties are also automatically added to the Northbound IMO. This enhances the default functionality of Cisco Prime Network. In addition to this, you can assign various types of alarm conditions to soft properties.

All property definitions and parameters are maintained in XML metadata in the registry. Figure 5-9 shows a high-level overview of how BQL can be used to manage Soft Properties.

Figure 5-9 Managing Soft Properties Using BQL

283540



By default, Prime Network VNEs model a subset of the device properties, which cover the most important and commonly used properties. Prime Network offers the Soft Properties mechanism to enable user-configurable extension of device modeling, which can cover any unsupported MIB variable. To view the changes, you must restart the VNE or unit for the soft property publishing to take effect.

Every Soft Property is implemented through a set of definitions that determine how to retrieve, parse, and display a certain MIB variable from the NE. Soft properties are retrieved from the NE using SNMP or Telnet/SSH.

Based on the defined retrieval specification, the soft property details are retrieved. Also, depending on the polling interval specified for the soft property, the soft property details may take time to displays in the Prime Network inventory.
With alarm thresholding mechanism, Prime Network monitors selected properties and generates an alarm every time these properties cross a user-defined threshold or violate a condition. This eliminates the need for OSS applications to constantly upload huge amounts of data and process it. Instead, Prime Network filters out irrelevant data, and sends only meaningful notifications.

Additional Reading

- Review *Cisco Prime Network 4.2.2 Administrator Guide* to understand the Prime Network user roles and scopes.
- Review *Cisco Prime Network 4.2.2 Customization Guide* to understand the Soft Properties and Alarm Threshold mechanisms.
- See the *Cisco Prime Network Information Model* Javadoc to understand the IMO for soft property. This document is available on the Prime Network Technology

Center website. You must have a Cisco.com account with partner level access, or you must be a Prime Network licensee to access this website.

5.7.1 Soft Property Interfaces

Table 5-13 lists the BQL commands supported for Soft Property queries.

Table 5-13 Supported Soft Property BQL Command Queries

Command Name	IMO/OID Type	IMO/OID Value	Description
GetAllSoftPropertiesForIMO	com.sheer.imo.keys.IManagedElementOid	{[ManagedElement(Key=*DeviceName*)]}	Retrieves all soft properties details for an NE.
GetPublished ElementVersions	com.sheer.imo.keys.ISoftPropertyOid	{[SoftProperty(ContextImoType=*SoftPropertyInterface*)(Name=*NameofSoftProperty*)(RegistryPat h=*SoftPropertyPath*)]}	Retrieves all versions of available soft properties for an NE.
Get	com.sheer.imo.keys.ISoftPropertyOid	{[SoftProperty(ContextImoType=*SoftPropertyInterface*)(Name=*NameofSoftProperty*)(RegistryPat h=*SoftPropertyPath*)]} Where, • SoftPropertyInterface: The interface type where the soft property was added. For example, if you have added the soft property to a port, the IMO value is com.sheer.imo.IPortConnector. • SoftPropertyPath: The path where the soft property is created. If you have published the soft property, by default, the published soft property is under “site/agentdefaults/da” directory.	Retrieves the specified soft property details.

Table 5-14 lists the BQL commands supported for Soft Property operations.

Table 5-14 Supported Soft Property BQL Command Operations

Command Name	IMO/OID Type	IMO/OID Value	Description
Set	com.sheer.imo.keys.ISoftPropertyOid	{[SoftProperty(ContextImoType=*com.sheer.imo.IManagedElement*)(Name=*NameofSoftProperty*)]}	Creates or edits a soft property.
Delete	com.sheer.imo.keys.ISoftPropertyOid	{[SoftProperty(ContextImoType=*SoftPropertyInterface*)(Name=*NameofSoftProperty*)(RegistryPath=*SoftPropertyPath*)]}	Deletes a soft property.

5.7.2 Samples BQL Scripts for Soft Property

This section contains the following sample BQL scripts:

- Creating a Soft Property with Parsing and Alarm Threshold, page 199
- Creating a Soft Property without Parsing and Alarm Threshold, page 201
- Retrieving All Soft Properties Details for an NE, page 201
- Deleting a Soft Property, page 204

Creating a Soft Property with Parsing and Alarm Threshold

The following example shows the usage of the BQL **Set** command to create a soft property (name: sp1;label: sp1) with Parsing and Alarm Threshold (tca1) parameters for an NE (c7-sw6). In this example,

- Command that is run on the NE is `show clock`.
- Alarm is triggered when the value is above 4 and it is cleared when the value is below 3.
- Alarm severity is assigned to critical.
- Alarm cannot be correlated to any other alarm; any other alarm cannot correlate to this alarm.
- Alarm is enabled for this soft property.
- Parsing type is Substring with starting index value as 4 and length as 5.
- Polling type as topo_l2 on the NE c7-sw6.
- Protocol used to connect to the NE is Telnet (1).
- Soft property is enabled on the NE.
- Param name *replace* set to *false* indicates that if the soft property already exist, it should not be replaced.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Set">
  <param name="imo">
        <value>
              <ISoftProperty>
                    <ID
```

```
type="Oid">{[SoftProperty(ContextImoType=com.sheer.imo.IManagedElement)(Na
me=sp1)]}</ID>
<Alarms type="IMObjects_Array">
                    <ISoftPropertyAlarm>
               <ID type="Oid">{[SoftPropertyAlarm(AlarmName=tca1)]}</ID>
<AlarmArguments type="IMObjects_Array">
                           <ISoftPropertyAlarmStringArgument>
                                 <ID
type="Oid">{[SoftPropertyAlarmArgument(ArgumentName=clear value)]}</ID>
<Value type="String">3</Value>
                           </ISoftPropertyAlarmStringArgument>
                           <ISoftPropertyAlarmStringArgument>
<ID
type="Oid">{[SoftPropertyAlarmArgument(ArgumentName=alarm value)]}</ID>
<Value type="String">4</Value>
                        </ISoftPropertyAlarmStringArgument>
                        </AlarmArguments>
                        <AlarmDescription type="String">tca alarm
                     example</AlarmDescription>
<AlarmSeverity type="String">CRITICAL</AlarmSeverity>
<CanBeCorrelated type="Boolean">false</CanBeCorrelated>
<CanCorrelate type="Boolean">false</CanCorrelate>
<Enabled type="Boolean">true</Enabled>
                           <TriggerEnum type="Integer">3</TriggerEnum>
                     </ISoftPropertyAlarm>
                  </Alarms>
               <CommandLine type="String">show clock</CommandLine>
                     <Description type="String" />
               <Enabled type="Boolean">true</Enabled>
               <Label type="String">sp1</Label>
                     <ParsingRules type="IMObjects_Array">
                           <ISoftPropertyParsingRule>
                  <ID type="Oid">{[SoftPropertyParsingRule(Index=0)]}</ID>
                           <InputBuffer type="String" />
                  <OperationEnum type="Integer">7</OperationEnum>
                           <OutputBuffer type="String" />
                           <ParsingArguments type="IMObjects_Array">
                                 <ISoftPropertyParsingRuleIntArgument>
                                 <ID
type="Oid">{[SoftPropertyParsingRuleArgument(ArgumentName=fromIndex)]}</ID
>
                     <Value type="Integer">4</Value>
                                 </ISoftPropertyParsingRuleIntArgument>
                                 <ISoftPropertyParsingRuleIntArgument>
                                       <ID
type="Oid">{[SoftPropertyParsingRuleArgument(ArgumentName=toIndex)]}</ID>
<Value type="Integer">5</Value>
                           </ISoftPropertyParsingRuleIntArgument>
                        </ParsingArguments>
                     </ISoftPropertyParsingRule>
                  </ParsingRules>
               <Polling type="String">topo_l2</Polling>
                     <ProtocolEnum type="Integer">1</ProtocolEnum>
                           <TableColumns type="IMObjects_Array" />
                           <TypeEnum type="Integer">1</TypeEnum>
                     </ISoftProperty>
                  </value>
```

```
              </param>
              <param name="neOid">
                    <value>{[ManagedElement(Key=c7-sw6)]}</value>
              </param>
              <param name="replace">
                    <value>false</value>
              </param>
</command>
.
```

Creating a Soft Property without Parsing and Alarm Threshold

The following example shows the usage of the BQL **Set** command to create a soft property (name: sp2; label: sp2label) for an NE (c7-sw6). In this example,

- Command that is run on the NE is `show clock`.
- Polling type as buffering on the NE c7-sw6.
- Protocol used to connect to the NE is Telnet (1).
- Soft property is enabled on the NE.
- Param name *replace* set to *false* indicates that if the soft property already exist, it should not be replaced.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Set">
  <param name="imo">
        <value>
              <ISoftProperty>
                    <ID
type="Oid">{[SoftProperty(ContextImoType=com.sheer.imo.IManagedElement)(Na
me=sp2)]}</ID>
                    <Alarms type="IMObjects_Array" />
                    <CommandLine type="String">show clock</CommandLine>
                    <Description type="String">description</Description>
                    <Enabled type="Boolean">true</Enabled>
                    <Label type="String">sp2label</Label>
                    <ParsingRules type="IMObjects_Array" />
                    <Polling type="String">buffering</Polling>
                    <ProtocolEnum type="Integer">1</ProtocolEnum>
                    <TableColumns type="IMObjects_Array" />
                    <TypeEnum type="Integer">1</TypeEnum>
                </ISoftProperty>
          </value>
  </param>
  <param name="neOid">
        <value>{[ManagedElement(Key=c7-sw6)]}</value>
  </param>
  <param name="replace">
        <value>false</value>
  </param>
</command>
.
```

Retrieving All Soft Properties Details for an NE

The following example shows the usage of the BQL **GetAllSoftPropertiesForIMO** command to retrieve all soft properties for the managed NE (ASR1004), which is of type Cisco ASR 1004) in Prime Network.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="GetAllSoftPropertiesForIMO">
  <param name="ne">
        <value>
          <IManagedElement>
            <ID type="Oid">{[ManagedElement(Key=ASR1004)]}</ID>
            <Aspects type="IMObjects_Array">
              <ISeverityAspect>
                  <ID
type="Oid">{[ManagedElement(Key=ASR1004)][SeverityAspect]}</ID>
                  <ClearedNum type="Integer">0</ClearedNum>
                  <CriticalNum type="Integer">0</CriticalNum>
                  <IndeterminateNum type="Integer">0</IndeterminateNum>
                  <InfoNum type="Integer">0</InfoNum>
                  <MajorNum type="Integer">0</MajorNum>
                  <MinorNum type="Integer">0</MinorNum>
                  <NotAckedClearedNum
                  type="Integer">0</NotAckedClearedNum>
                  <NotAckedCriticalNum
                  type="Integer">0</NotAckedCriticalNum>
                  <NotAckedIndeterminateNum
                  type="Integer">0</NotAckedIndeterminateNum>
                  <NotAckedInfoNum type="Integer">0</NotAckedInfoNum>
                  <NotAckedMajorNum type="Integer">0</NotAckedMajorNum>
                  <NotAckedMinorNum type="Integer">0</NotAckedMinorNum>
                  <NotAckedPropClearedNum
                  type="Integer">0</NotAckedPropClearedNum>
                  <NotAckedPropCriticalNum
                  type="Integer">0</NotAckedPropCriticalNum>
                  <NotAckedPropIndeterminateNum
type="Integer">0</NotAckedPropIndeterminateNum>
                  <NotAckedPropInfoNum
                  type="Integer">0</NotAckedPropInfoNum>
                  <NotAckedPropMajorNum
                  type="Integer">1</NotAckedPropMajorNum>
                  <NotAckedPropMinorNum
                  type="Integer">0</NotAckedPropMinorNum>
                  <NotAckedPropWarningNum
                  type="Integer">0</NotAckedPropWarningNum>
                  <NotAckedWarningNum
                  type="Integer">0</NotAckedWarningNum>
                  <PropClearedNum type="Integer">0</PropClearedNum>
                  <PropCriticalNum type="Integer">0</PropCriticalNum>
                  <PropIndeterminateNum
                  type="Integer">0</PropIndeterminateNum>
                  <PropInfoNum type="Integer">0</PropInfoNum>
                  <PropMajorNum type="Integer">1</PropMajorNum>
                  <PropMinorNum type="Integer">0</PropMinorNum>
                  <PropWarningNum type="Integer">0</PropWarningNum>
                  <WarningNum type="Integer">0</WarningNum>
              </ISeverityAspect>
              <newalarm.ITicketListAspect>
                    <ID
```

```
type="Oid">{[ManagedElement(Key=ASR1004)][TicketListAspect]}</ID>
                    <Tickets type="IMObjects_Array">
                      <newalarm.ITicket>
                           <ID type="Oid">{[NewAlarm(Id=1)]}</ID>
                             <AffectedDevicesCount
                           type="Integer">1</AffectedDevicesCount>
                             <AggregatedAckStateEnum
type="Integer">0</AggregatedAckStateEnum>
                           <AggregatedSeverityEnum
                           type="Integer">5</AggregatedSeverityEnum>
                           <AlarmCount type="Integer">1</AlarmCount>
                           <Archived type="Boolean">false</Archived>
                           <AutoCleared type="Boolean">false</AutoCleared>
                           <DuplicationCount
                           type="Integer">2</DuplicationCount>
                           <EventCount type="Integer">2</EventCount>
                           <LastModificationTime type="java.util.Date">Thu
                           May 06
                           15:40:54 IST 2010</LastModificationTime>
                           <LatestState type="String">Card
                           down</LatestState>
                           <ReductionCount
                           type="Integer">2</ReductionCount>
                           <SeverityEnum type="Integer">5</SeverityEnum>
                           <Source
                           type="Oid">{[ManagedElement(Key=ASR1004)][Physic
                           alRoot][Chassis][Slot(SlotNum=201)][Module][Slot
                           (SlotNum=15)][Module]}</Source>
                      </newalarm.ITicket>
                    </Tickets>
                  </newalarm.ITicketListAspect>
              </Aspects>
              <CommunicationStateEnum
              type="Integer">3</CommunicationStateEnum>
              <DeviceName type="String">ASR1004</DeviceName>
              <ElementCategoryEnum type="Integer">4</ElementCategoryEnum>
              <ElementType type="String">Cisco ASR 1004</ElementType>
              <ElementTypeKey type="String">CISCO_ASR_1004</ElementTypeKey>
              <IP type="com.sheer.types.IPAddress">10.77.214.140</IP>
              <InvestigationStateEnum
              type="Integer">11</InvestigationStateEnum>
              <ScriptMetadataOids type="com.sheer.framework.imo.Oid_Array">

<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscovdc-cisco-nccm-
cmds/config-commit)][
ScriptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>

<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscovdc-cisco-nccm-
cmds/config-deploy)][
ScriptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>

<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscovdc-cisco-nccm-
cmds/config-fetch)][S
criptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>
```

```
<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscovdc-cisco-nccm-
cmds/neim-activate-im age-
telnet)][ScriptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>

<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscovdc-cisco-nccm-
cmds/neim-delete-imag e-
snmp)][ScriptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>

<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscovdc-cisco-nccm-
cmds/neim-delete-imag e-
telnet)][ScriptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>

<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscovdc-cisco-nccm-
cmds/neim-distribute- activate-image-
telnet)][ScriptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>

<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscovdc-cisco-nccm-
cmds/neim-distribute- image-
snmp)][ScriptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>

<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscovdc-cisco-nccm-
cmds/neim-distribute- image-
telnet)][ScriptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>

<com.sheer.framework.imo.Oid>{[Command(CommandId=ciscovdc-cisco-nccm-
cmds/neim-verify-acti vate-
image)][ScriptMetadata(MetadataId=cisco)]}</com.sheer.framework.imo.Oid>
              </ScriptMetadataOids>
              <SoftwareVersion type="String">15.0(20100426:101448)
        [UNKNOWN BRANCH
-BLD-BLD_V150_1_S_XE31_THROTTLE_LATEST_20100426_080025-ios 116]
</SoftwareVersion>
              <SysDescription type="String">Cisco IOS Software, IOS-XE
        Software (PPC_LINUX_IOSD-ADVENTERPRISEK9-M), Experimental Version
        15.0(20100426:101448) [UNKNOWN BRANCH -BLD-
        BLD_V150_1_S_XE31_THROTTLE_LATEST_20100426_080025-ios 116]
Copyright (c) 1986-2010 by Cisco Systems, Inc. Compiled
Mo</SysDescription>
              <SysLocation type="String" />
              <SysName type="String">ASR1004.cisco</SysName>
              <SysUpTime type="java.util.Date">Mon May 03 20:29:16 IST
              2010</SysUpTime>
              <VendorEnum type="Integer">3</VendorEnum>
           </IManagedElement>
        </value>
  </param>
</command>
.
```

Deleting a Soft Property

The following example shows the usage of the BQL Delete command to delete the soft property sp1 on the NE (c7-sw6), belonging to AVM (123).

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Delete">
  <param name="oid">
```

```
    <value>{[SoftProperty(ContextImoType=com.sheer.imo.IPortConnector)(Na
  me=sp1)(RegistryPath=avm123/agents/da/c7-
  sw6/imo/registrations)]}</value>
  </param>
</command>
.
```

Soft Property with TCA Threshold with Negative Values

The following example shows the usage of the BQL **Set** command to define the TCA with negative value.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Set">
  <param name="imo">
        <value>
              <ISoftProperty>
                <ID
          type="Oid">{[SoftProperty(ContextImoType=com.sheer.imo.IManagedEl
          ement)(Name=description)(
          RegistryPath=avm444/agents/da/Roi1/imo/registrations)]}</ID>
                    <Alarms type="IMObjects_Array">
                          <ISoftPropertyAlarm>
                            <ID
                      type="Oid">{[SoftProperty(ContextImoType=com.sheer.i
                      mo.IManagedElement)(Name=description)(
                      RegistryPath=avm444/agents/da/Roi1/imo/registrations
                      )][SoftPropertyAlarm(AlarmName=thresho
                      ld_8)(AlarmType=9015)]}</ID>
                           <AlarmArguments type="IMObjects_Array">
                                 <ISoftPropertyAlarmStringArgument>
                                       <ID
type="Oid">{[SoftPropertyAlarmArgument(ArgumentName=clear value)]}</ID>
                                       <Value type="String">-10</Value>
                                 </ISoftPropertyAlarmStringArgument>
                                 <ISoftPropertyAlarmStringArgument>
                                       <ID
type="Oid">{[SoftPropertyAlarmArgument(ArgumentName=alarm value)]}</ID>
                                       <Value type="String">-3</Value>
                                 </ISoftPropertyAlarmStringArgument>
                           </AlarmArguments>
                           <AlarmDescription
                    type="String">threshold_8</AlarmDescription>
                           <AlarmSeverity
                           type="String">CRITICAL</AlarmSeverity>
                           <CanBeCorrelated
                           type="Boolean">false</CanBeCorrelated>
                           <CanCorrelate
                           type="Boolean">false</CanCorrelate>
                           <Enabled type="Boolean">true</Enabled>
                           <TriggerEnum type="Integer">3</TriggerEnum>
                    </ISoftPropertyAlarm>
<InputBuffer type="String" />
              <OperationEnum type="Integer">7</OperationEnum>
              <OutputBuffer type="String" />
              <ParsingArguments type="IMObjects_Array">
                    <ISoftPropertyParsingRuleIntArgument>
```

```
                        <IDtype="Oid">{[SoftPropertyParsingRuleArgument(
                    ArgumentName=fromIndex)]}</ID>
                        <Value type="Integer">15</Value>
                    </ISoftPropertyParsingRuleIntArgument>
                    <ISoftPropertyParsingRuleIntArgument>
                        <IDtype="Oid">{[SoftPropertyParsingRuleArgument(
                      ArgumentName=toIndex)]}</ID>
                            <Value type="Integer">17</Value>
                    </ISoftPropertyParsingRuleIntArgument>
                </ParsingArguments>
              </ISoftPropertyParsingRule>
            </ParsingRules>
            <Polling type="String">status</Polling>
            <ProtocolEnum type="Integer">1</ProtocolEnum>
            <TableColumns type="IMObjects_Array" />
            <TypeEnum type="Integer">1</TypeEnum>
        </ISoftProperty>
      </value>
  </param>
  <param name="neOid">
        <value>{[ManagedElement(Key=Roi1)]}</value>
  </param>
  <param name="replace">
        <value>true</value>
  </param>
</command>
```

5.8 Running Configuration Backup and Restore Operations Using BQL

Cisco Prime Network Change and Configuration Management (CCM) provides tools that allow you to manage the software and device configuration changes that are made to devices in network. Backup operation allows you to archive device configuration files in Prime Network. Archive configuration files can be restored back to device at a later time. The following sections provide a list of BQL commands supported for Configuration Backup, restore and export operations.

5.8.1 Configuration Backup and Restore Interfaces

Table 5-15 lists the BQL commands supported for Configuration Backup and Restore queries.

Table 5-15 Supported Configuration Backup & Restore BQL Command Queries

Command Name	IMO/OID Type	IMO/OID Value	Description
CreateCCMConfigBackupJob	ICCMDeviceWrapper ICCMBackupJobParams IJobTriggerParams	{[CCMDeviceWrapper]} {[CCMBackupJobParams]} {[JobTriggerParams]}	Creates Configuration Backup Job
CreateCCMConfigRestoreJob	ICCMConfigArchivesWrapper ICCMRestoreJobParams IJobTriggerParams	{[CCMConfigArchivesWrapper]} {[CCMRestoreJobParams]} {[JobTriggerParams]}	Creates Configuration Restore Job
CreateCCMConfigExportJob	ICCMConfigArchivesWrapper ICCMExportJobParams, IJobTriggerParams	{[CCMConfigArchivesWrapper]} {[CCMExportJobParams]} {[JobTriggerParams]}	Creates Configuration Archive Export Job
GetCCMJobs	–	–	Retrieves CCM Job details.

5.8.2 Sample BQL Scripts for a Configuration Backup and Restore Operations

This section contains the following sample BQL scripts:

- Create Configuration Backup Job, page 207
- Create Configuration Restore Job, page 208
- Create Configuration Export Job, page 209
- Get CCM Job Details, page 210

5.8.2.1 Create Configuration Backup Job

The following example shows the usage of the BQL **CreateCCMConfigBackupJob** command to create a Configuration Backup Job.

```
<command name="CreateCCMConfigBackupJob">
<param name="devices">
   <value>
       <scheduler.ICCMDeviceWrapper type="scheduler.ICCMDeviceWrapper"
instance_id="0">
          <ManagedElementOid
type="String">{[ManagedElement(Key=10.56.59.23)]}</ManagedElementOid>
       </scheduler.ICCMDeviceWrapper>
   </value>
</param>
```

```
<param name="backupparams">
<value>
     <scheduler.ICCMBackupJobParams type="scheduler.ICCMBackupJobParams"
instance_id="0">
        <Transferprotocol type="String">TFTP</Transferprotocol>
        <Description type="String">Test NBI Backup</Description>
        <DeviceUserName type="String"></DeviceUserName>
        <DevicePassword type="String"></DevicePassword>
     </scheduler.ICCMBackupJobParams>
</value>
</param>
<param name="triggerparams">
<value>
        <scheduler.IJobTriggerParams type="scheduler.IJobTriggerParams"
instance_id="0">
             <TriggerType type="String">AS_SOON_AS</TriggerType>
             <RunIndefinite type="String"></RunIndefinite>
             <RepeatInterval type="String"></RepeatInterval>
             <StartDate type="String"></StartDate>
             <StartTime type="String"></StartTime>
             <DaysofWeek type="String"></DaysofWeek>
             <DaysofMonth type="String"></DaysofMonth>
        </scheduler.IJobTriggerParams>
</value>
</param>
</command>
.
```

Command Response

```
<JobDetails>
        <JobName>ConfigMgmt-Backup-d022374f-c76b-4d29-90d4-
8a2f37e27e5e$$$05-31-2013 11:36:06</JobName>
        <JobSpecId>74074</JobSpecId>
</JobDetails>
```

5.8.2.2 Create Configuration Restore Job

The following example shows the usage of the BQL **CreateCCMConfigRestoreJob** command to create a Configuration Restore Job.

```
<command name="CreateCCMConfigRestoreJob">
<param name="archives">
   <value>
      <scheduler.ICCMConfigArchivesWrapper
type="scheduler.ICCMConfigArchivesWrapper" instance_id="0">
         <ManagedElementOid
type="String">{[ManagedElement(Key=10.56.59.23}</ManagedElementOid>
         <ConfigType type="String">startup</ConfigType>
         <Ver type="String">1</Ver>
         <Contextname type="String">N/A</Contextname>
         <CommitId type="String"></CommitId>
       </scheduler.ICCMConfigArchivesWrapper>
   </value>
</param>
<param name="restoreparams">
  <value>
```

```
     <scheduler.ICCMRestoreJobParams type="scheduler.ICCMRestoreJobParams"
instance_id="0">
         <Transferprotocol type="String">TFTP</Transferprotocol>
         <Description type="String">nbi_restore</Description>
         <RestoreOptions type="String">running</RestoreOptions>
         <CfgRestoreMode type="String">merge</CfgRestoreMode>
         <EnableFallback type="String">no</EnableFallback>
         <DeviceUserName type="String"></DeviceUserName>
         <DevicePassword type="String"></DevicePassword>
     </scheduler.ICCMRestoreJobParams>
  </value>
</param>
<param name="triggerparams">
<value>
        <scheduler.IJobTriggerParams type="scheduler.IJobTriggerParams"
instance_id="0">
           <TriggerType type="String">AS_SOON_AS</TriggerType>
                <RunIndefinite type="String">false</RunIndefinite>
                <RepeatInterval type="String"></RepeatInterval>
                <StartDate type="String"></StartDate>
                <StartTime type="String"></StartTime>
                <DaysofWeek type="String"></DaysofWeek>
                <DaysofMonth type="String"></DaysofMonth>
        </scheduler.IJobTriggerParams>
</value>
</param>
</command>
.
```

Command Response

```
<JobDetails>
        <JobName>ConfigMgmt-Restore-9be459fc-43f1-4a20-8ec1-
6c95188fb8e0$$$05-31-2013 11:42:11</JobName>
        <JobSpecId>74075</JobSpecId>
</JobDetails>
```

5.8.2.3 Create Configuration Export Job

The following example shows the usage of the BQL **CreateCCMConfigExportJob** command to create a Configuration Export Job.

```
<command name="CreateCCMConfigExportJob">
<param name="archives">
    <value>
        <scheduler.ICCMConfigArchivesWrapper
type="scheduler.ICCMConfigArchivesWrapper" instance_id="0">
           <ManagedElementOid
type="String">{[ManagedElement(Key=N5K)]}</ManagedElementOid>
           <ConfigType type="String">running</ConfigType>
           <Ver type="String">5</Ver>
           <Contextname type="String">N/A</Contextname>
           <CommitId type="String"></CommitId>
           <IsEditedVersion type="String">false</IsEditedVersion>
           <EditedVersionNumber type="String">0</EditedVersionNumber>
        </scheduler.ICCMConfigArchivesWrapper>
     </value>
```

```
</param>
<param name="exportparams">
<value>
        <scheduler.ICCMExportJobParams
type="scheduler.ICCMExportJobParams" instance_id="0">
          <Description type="String">nbi_export</Description>
        </scheduler.ICCMExportJobParams>
</value>
</param>
<param name="triggerparams">
<value>
        <scheduler.IJobTriggerParams type="scheduler.IJobTriggerParams"
instance_id="0">
                <TriggerType type="String">AS_SOON_AS</TriggerType>
                <RunIndefinite type="String">false</RunIndefinite>
                <RepeatInterval type="String"></RepeatInterval>
                <StartDate type="String"></StartDate>
                <StartTime type="String"></StartTime>
                <DaysofWeek type="String"></DaysofWeek>
                <DaysofMonth type="String"></DaysofMonth>
        </scheduler.IJobTriggerParams>
</value>
</param>
</command>
.
```

Command Response

```
<JobDetails>
        <JobName>ConfigMgmt-Export-bb73daf7-5fd3-4627-b63d-
3a5c1ca5066c$$$02-24-2013 07:56:59</JobName>
        <JobSpecId>721721</JobSpecId>
</JobDetails>
```

5.8.2.4 Get CCM Job Details

The following example shows the usage of the BQL **GetCCMJob** command to retrieve CCM Job details.

```
<command name="GetCCMJob">
<param name="jobspecid">
    <value>74074</value>
</param>
</command>
.
```

Command Response

```
<JobInformation>
        <JobSpecID>74074</JobSpecID>
        <LastRunDetails>
                <JobID>71073</JobID>
                <JobType>ConfigMgmt-Backup</JobType>
                <ScheduledTime>Fri May 31 11:36:06 IDT
2013</ScheduledTime>
                <CompletedTime>Fri May 31 11:36:56 IDT
2013</CompletedTime>
                <Status>COMPLETED</Status>
                <Result>SUCCESS</Result>
```

```
                <Owner>root</Owner>
                <Comments>test nbi backup</Comments>
                <SuccessfulTasks>
                        <DeviceName> 10.56.59.23</DeviceName>
                        <Summary> Device: Config Backup Operation
Completed.

Supported Configuration Types:

RUNNING_CONFIG
STARTUP_CONFIG

RUNNING_CONFIG - Device Config is in Sync with Latest Archive Version.

STARTUP_CONFIG - Device Config is in Sync with Latest Archive Version.
                        </Summary>

                </SuccessfulTasks>
        </LastRunDetails>
</JobInformation>
```

5.9 Running Compliance Audit Using BQL

The Compliance Audit feature allows you to ensure that existing device configurations comply with their deployment's policies. The Configuration Audit feature allows you to perform Compliance Audit on network devices to identify the device that does not comply with specific policies. The operation identifies non-compliant devices and provides violation reports per device. You can perform Compliance fix operation to fix violations identified by the Compliance Audit operation. The following sections provide a list of BQL commands supported for Compliance Audit and Fix operations.

5.9.1 Compliance Audit Interfaces

Table 5-16 lists the BQL commands supported for Configuration Audit operations.

Table 5-16 Supported Configuration Audit BQL Commands

Command Name	IMO/OID Type	IMO/OID Value	Description
GetAllPolicyProfiles	–	–	Retrieves the policy profiles from Complaince Manager
CreateComplianceAuditJob	IComplianceAuditParams IDeviceWrapper IJobTriggerParams	{[ComplianceAuditParams]} {[DeviceWrapper]} {[JobTriggerParams]}	Creates Compliance Audit Job
GetComplianceAuditJobResult	–	–	Retrieves the Compliance Audit job result.

Command Name	IMO/OID Type	IMO/OID Value	Description
CreateComplianceFixJob	IComplianceFixParams IDeviceWrapper IJobTriggerParams IComplianceViolation IComplianceInput	{[ComplianceFixParams]} {[DeviceWrapper]} {[JobTriggerParams]}	Creates Compliance Violation Fix Job
GetComplianceFixJobResult	–	–	Retrieves the result of Compliance fix job

5.9.2 Sample BQL Commands for Complaince Audit Operations

This section contains the following sample BQL commands:

- Get All Policy Profiles
- Create Compliance Audit Job
- Get Complaince Audit Job Result
- Create Compliance Fix Job
- Get Compliance Fix Job Result

5.9.2.1 Get All Policy Profiles

The following example shows the usage of the BQL **GetAllPolicyProfiles** command to retrieve all the Policy Profiles from Prime Network.

```
<command name="GetAllPolicyProfiles">
</command>
.
```

Command Response

```
<PolicyProfiles>
    <PolicyProfile>
      <Name>WWPG</Name>
      <Description>WWPG</Description>
    </PolicyProfile>
    <PolicyProfile>
      <Name>TestPGStar</Name>
      <Description></Description>
    </PolicyProfile>
    <PolicyProfile>
      <Name>WW</Name>
      <Description></Description>
    </PolicyProfile>
  </PolicyProfiles>
</CommandResult>
```

5.9.2.2 Create Compliance Audit Job

The following example shows the usage of the BQL **CreateComplianceAuditJob** command to create a Compliance Audit Check Job.

```
<command name="CreateComplianceAuditJob">
  <param name="auditparams">
    <value>
      <compliance.IComplianceAuditParams
type="compliance.IComplianceAuditParams" instance_id="0">
        <Comments type="String">FixInput Test</Comments>
        <EmailId type="String">kpachiap@cisco.com</EmailId>
        <DeviceUserName type="String">cisco</DeviceUserName>
        <DevicePassword type="String">cisco</DevicePassword>
        <FetchRunningConfigFrom
type="String">LATEST_FROM_DEVICE</FetchRunningConfigFrom>
        <PolicyProfileName type="String">FixInput</PolicyProfileName>
        <JobName type="String">FixInputHelloWorld</JobName>
      </compliance.IComplianceAuditParams>
    </value>
  </param>
  <param name="devices">
    <value>
      <compliance.IDeviceWrapper type="compliance.IDeviceWrapper"
instance_id="0">
        <ManagedElementOid type="String">{[ManagedElement(Key=IOS-
7609s)]}</ManagedElementOid>
      </compliance.IDeviceWrapper>
      <compliance.IDeviceWrapper type="compliance.IDeviceWrapper"
instance_id="1">
        <ManagedElementOid type="String">{[ManagedElement(Key=Nexus-
5k)]}</ManagedElementOid>
      </compliance.IDeviceWrapper>
    </value>
  </param>
  <param name="triggerparams">
    <value>
      <scheduler.IJobTriggerParams type="scheduler.IJobTriggerParams"
instance_id="0">
        <TriggerType type="String">AS_SOON_AS</TriggerType>
      </scheduler.IJobTriggerParams>
    </value>
  </param>
</command>
```

. Command Response

```
<CommandResult>
  <JobDetails>
    <JobName>Compliance-Audit-c4ad85b1-215a-46ba-9247-
8069110a00f1$$$05-29-2013 01:34:11</JobName>
    <JobSpecId>61075</JobSpecId>
  </JobDetails>
</CommandResult>
```

5.9.2.3 Get Complaince Audit Job Result

The following example shows the usage of the BQL **GetComplianceAuditJobResult** command to retrieve results of a Compliance Audit Job.

```
<command name="GetComplianceAuditJobResult">
<param name="jobspecid">
    <value>61075</value>
</param>
</command>
.
```

Command Response

```
<ComplianceAuditResult>
  <JobInfo>
    <StartTime>Fri Jan 10 2014 18:23:24 IST</StartTime>
    <EndTime>Fri Jan 10 2014 18:24:34 IST</EndTime>
    <PolicyProfileName>FixInputes</PolicyProfileName>
    <JobStatus>Completed</JobStatus>
    <JobResult>Failure</JobResult>
    <Comments>Testing</Comments>
    <JobSpecId>315316</JobSpecId>
    <JobOwner>root</JobOwner>
    <JobType>Compliance-Audit</JobType>
    <JobId>319320</JobId>
    <AssociatedPolicyCount>1</AssociatedPolicyCount>
    <AuditedDeviceCount>1</AuditedDeviceCount>
    <NonAuditedDeviceCount>0</NonAuditedDeviceCount>
    <ScheduledAt>Fri Jan 10 2014 18:23:24 IST</ScheduledAt>
    <CompletedAt>Fri Jan 10 2014 18:24:34 IST</CompletedAt>
  </JobInfo>
  <JobDetails>
    <ViolationDevices>
      <Device>
        <DeviceName>{[ManagedElement(Key=IOS-7609s)]}</DeviceName>
        <DeviceViolationCount>1 Violation(s)</DeviceViolationCount>
        <DeviceMaxSeverity>Minor</DeviceMaxSeverity>
        <DeviceFixable>true</DeviceFixable>
        <PolicyInfo>
          <PolicyName>FixInputs</PolicyName>
          <PolicyViolationCount>1 Violation(s)</PolicyViolationCount>
          <PolicyMaxSeverity>Minor</PolicyMaxSeverity>
          <PolicyFixable>true</PolicyFixable>
          <ViolationsInfo>
            <Violation>
              <ViolationId>322323</ViolationId>
              <ViolationSeverity>Minor</ViolationSeverity>
              <ViolationFixable>true</ViolationFixable>
              <ViolationMessage>fix input test</ViolationMessage>
              <FixInputs>
                <Input id="_mask" type="IpMask"
required="true"></Input>
                <Input id="_string" type="String" required="true"
default="ddddd"></Input>
                <Input id="_interface" type="Interface"
required="true"></Input>
```

```
                <Input id="_int" type="Integer" required="true"
minInclusive="12"  maxInclusive="45"></Input>
                <Input id="_stringLov" type="String" required="true"
default="1">
                  <Options>
                    <Option id="1">One</Option>
                    <Option id="2">Two</Option>
                  </Options>
                </Input>
                <Input id="_ip" type="IpAddress" required="true"
default="1.2.3.4"></Input>
            </FixInputs>
          </Violation>
        </ViolationsInfo>
      </PolicyInfo>
    </Device>
  </ViolationDevices>
</JobDetails>
</ComplianceAuditResult>
```

5.9.2.4 Create Compliance Fix Job

The following example shows the usage of the BQL **CreateComplianceFixJob** command to create a Compliance Violation Fix Job.

```
<command name="CreateComplianceFixJob">
  <param name="fixparams">
    <value>
      <compliance.IComplianceFixParams
type="compliance.IComplianceFixParams" instance_id="0">
        <Comments type="String">Testing</Comments>
        <EmailId type="String">shmuthur@cisco.com</EmailId>
        <DeviceUserName type="String">cisco</DeviceUserName>
        <DevicePassword type="String">cisco</DevicePassword>
        <JobSpecId type="String">281282</JobSpecId>
        <JobName type="String">Fix Test</JobName>
      </compliance.IComplianceFixParams>
    </value>
  </param>
  <param name="devices">
    <value>
      <compliance.IDeviceWrapper type="compliance.IDeviceWrapper"
instance_id="0">
        <ManagedElementOid type="String">{[ManagedElement(Key=Nexus-
5k)]}</ManagedElementOid>
        <Violation type="IMObjects_Array">
          <compliance.IComplianceViolation
type="compliance.IComplianceViolation" instance_id="1">
            <Id type="String">290291</Id>
            <FixInputValue type="IMObjects_Array">
              <compliance.IComplianceInput
type="compliance.IComplianceInput" instance_id="1">
                <Id type="String">_B</Id>
                <Value type="String">10</Value>
              </compliance.IComplianceInput>
              <compliance.IComplianceInput
type="compliance.IComplianceInput" instance_id="2">
```

```
                    <Id type="String">_A</Id>
                    <Value type="String">11</Value>
                  </compliance.IComplianceInput>
                </FixInputValue>
              </compliance.IComplianceViolation>
            </Violation>
          </compliance.IDeviceWrapper>
        </value>
      </param>
      <param name="triggerparams">
        <value>
          <scheduler.IJobTriggerParams type="scheduler.IJobTriggerParams"
instance_id="0">
            <TriggerType type="String">AS_SOON_AS</TriggerType>
          </scheduler.IJobTriggerParams>
        </value>
      </param>
    </command>
```

Command Response

```
    <CommandResult>
      <JobDetails>
        <JobName>Compliance-Fix-37468396-1412-42bc-bbc9-69e13f72144a$$$05-
29-2013 01:30:49</JobName>
        <JobSpecId>61074</JobSpecId>
      </JobDetails>
    </CommandResult>
```

5.9.2.5 Get Compliance Fix Job Result

The following example shows the usage of the BQL **GetComplianceFixJobResult** command to retrieve results of a Compliance Fix Job.

```
    <command name="GetComplianceFixJobResult">
    <param name="jobspecid">
        <value>61074</value>
    </param>
    </command>
    .
```

Command Response

```
    <ComplianceFixResult>
      <JobInfo>
        <StartTime>Fri Jan 10 2014 18:29:39 IST</StartTime>
        <EndTime>Fri Jan 10 2014 18:30:10 IST</EndTime>
        <JobStatus>Completed</JobStatus>
        <JobResult>Success</JobResult>
        <Comments></Comments>
        <ScheduledAt>Fri Jan 10 2014 18:29:39 IST</ScheduledAt>
        <CompletedAt>Fri Jan 10 2014 18:30:10 IST</CompletedAt>
      </JobInfo>
      <JobDetails>
        <FixResult>
          <Device>
            <DeviceName>{[ManagedElement(Key=ASR5K-StarOS)]}</DeviceName>
            <ViolationId>322325</ViolationId>
```

```
          <Status>Success</Status>
        </Device>
        <Device>
          <DeviceName>{[ManagedElement(Key=IOS-7609s)]}</DeviceName>
          <ViolationId>322326</ViolationId>
          <Status>Failure</Status>
          <Reason>Unable to push the Fix CLI :: FAILED - Failed to deploy
running-config to device.</Reason>
        </Device>
        <Device>
          <DeviceName>{[ManagedElement(Key=Nexus-5k)]}</DeviceName>
          <ViolationId>322324</ViolationId>
          <Status>Success</Status>
        </Device>
      </FixResult>
    </JobDetails>
  </ComplianceFixResult>
```

5.10 Running Transactions Using BQL

Transaction Manager provides a framework that can be used to schedule and run *transactions* (workflows) that are created using the Prime Network XDE Eclipse SDK.

Transactions are basically XDE procedures that contain tasks that are grouped together and specified into a flow, with certain sequences, branches, and failure policies (including rollback procedures). Transactions can include command scripts that are created using Command Manager. The same holds true for Command Builder as long as the transaction was created using the Prime Network XDE Eclipse SDK (which has special features to work with Command Builder scripts). The following sections provide a list of BQL commands supported by Transaction Manager to run transactions.

5.10.1 Transaction Manager Interfaces

Table 5-17 lists the BQL commands supported for Transaction Manager operations.

Table 5-17 Supported Transaction Manager BQL Commands

Command Name	IMO/OID Type	IMO/OID Value	Description
GetAllTransactions	-	-	Retrieves all the Transactions available on the server
GetTransactionInputParameters	com.sheer.imo.scheduler.ITransactionDetail	{[TransactionDetail]}	Retrieves all the input parameters of a specific Transaction

Command Name	IMO/OID Type	IMO/OID Value	Description
CreateTransactionManagerJob	com.sheer.imo.scheduler.ITransactionDetail com.sheer.imo.scheduler.ITransactionManagerDeviceWrapper com.sheer.imo.scheduler.ITransactionManagerJobInputParameter com.sheer.imo.scheduler.IJobTriggerParams com.sheer.imo.scheduler.ITransactionManagerJobInputParameter	{[TransactionDetail]} {[TransactionManagerDeviceWrapper]} {[TransactionManagerJobInputParameter]} {[JobTriggerParams]} {[TransactionManagerJobInputParameter]}	Creates transaction manager job
Reschedule	com.sheer.imo.keys.IJobOid com.sheer.imo.scheduler.ITransactionManagerJobInputParameter	{[Job(Name=transaction_manager_job_name)]} {[JobTriggerParams]}	Reschedules transaction manager job
GetTransactionManagerJobByFilter	com.sheer.imo.scheduler.IJobFilter	{[JobFilter]}	Retrieves all the transaction manager job(s) that matches the filter criteria
GetTransactionManagerJobResult	com.sheer.imo.keys.IJobOid	{[Job(Name=transaction_manager_job_name)]}	Retrieves the details of a transaction manager job
GetTransactionManagerTaskResult	—	—	Retrieves the result of a specific transaction manager task
DeleteJob	com.sheer.imo.keys.IJobOid	{[Job(Name=transaction_manager_job_name)]}	Deletes transaction manager job
UpdateJob	com.sheer.imo.keys.IJobOid	{[Job(Name=transaction_manager_job_name)]}	Suspend, resume or cancel transaction manager job based on the operation type passed.

5.10.2 Sample BQL Commands for a Transaction Manager Operations

This section contains the following sample BQL commands:

5.10.2.1 Get All Transactions

The following command example shows the usage of the BQL **GetAllTransactions** command to retrieve all the available Transactions within the Prime Network server from the XDE-home/standard folder. The values that are retrieved are Package Id, Transaction Name and its Description.

```
<command name="GetAllTransactions"
/>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionDetailWrapper type="scheduler.ITransactionDetailWrapper"
instance_id="0">
 <Status type="String">Success</Status>
 <TransactionDetails type="IMObjects_Array">
  <scheduler.ITransactionDetail type="scheduler.ITransactionDetail"
instance_id="1">
    <Description type="String">this is execute cb script</Description>
    <PackageID type="String">Demo</PackageID>
    <TransactionName type="String">ExecuteCBScript</TransactionName>
 </scheduler.ITransactionDetail>
 <scheduler.ITransactionDetail type="scheduler.ITransactionDetail"
instance_id="2">
   <Description type="String">test description</Description>
```

```
    <PackageID type="String">Demo</PackageID>
    <TransactionName type="String">procedure</TransactionName>
   </scheduler.ITransactionDetail>
  </TransactionDetails>
</scheduler.ITransactionDetailWrapper>
```

5.10.2.2 Get Transaction Input Parameters

The following command example shows the usage of the BQL **GetTransactionInputParameters** command to retrieve all the input params of a specific Transaction within the Prime Network server.

```
<command name="GetTransactionInputParameters">
 <param name="transactionDetails">
  <value>
   <scheduler.ITransactionDetail type="scheduler.ITransactionDetail"
instance_id="3">
      <Description type="String"></Description>
      <PackageID type="String">Set_of_Tests4</PackageID>
      <TransactionName type="String">Configure_Interface</TransactionName>
    </scheduler.ITransactionDetail>
  </value>
 </param>
 </command>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionDetail type="scheduler.ITransactionDetail"
instance_id="0">
  <InputParameters type="IMObjects_Array">
    <scheduler.ITransactionInputParameters
type="scheduler.ITransactionInputParameters" instance_id="1">
      <ParamName type="String">interface</ParamName>
    </scheduler.ITransactionInputParameters>
    <scheduler.ITransactionInputParameters
type="scheduler.ITransactionInputParameters" instance_id="2">
      <ParamName type="String">IPAddress</ParamName>
    </scheduler.ITransactionInputParameters>
  </InputParameters>
  <Status type="String">Success</Status>
</scheduler.ITransactionDetail>
```

5.10.2.3 Create Transaction Manager Job with same input params for all devices

The following command example shows the usage of the BQL **CreateTransactionManagerJob** command to create a transaction manager job with same input params for all devices

```
<command name="CreateTransactionManagerJob">
 <param name="transactionDetails">
  <value>
```

```
     <scheduler.ITransactionDetail type="scheduler.ITransactionDetail"
instance_id="4">
      <Description type="String"></Description>
      <PackageID type="String">NewProject</PackageID>
      <TransactionName type="String">executeCommand</TransactionName>
     </scheduler.ITransactionDetail>
    </value>
   </param>
   <param name="devices">
    <value>
     <scheduler.ITransactionManagerDeviceWrapper
type="scheduler.ITransactionManagerDeviceWrapper" instance_id="0">
      <ManagedElementOid type="Oid">{[ManagedElement(Key=c4-
upe1)]}</ManagedElementOid>
     </scheduler.ITransactionManagerDeviceWrapper>
     <scheduler.ITransactionManagerDeviceWrapper
type="scheduler.ITransactionManagerDeviceWrapper" instance_id="0">
      <ManagedElementOid type="Oid">{[ManagedElement(Key=c4-
upe2)]}</ManagedElementOid>
     </scheduler.ITransactionManagerDeviceWrapper>
     <scheduler.ITransactionManagerDeviceWrapper
type="scheduler.ITransactionManagerDeviceWrapper" instance_id="0">
      <ManagedElementOid type="Oid">{[ManagedElement(Key=c4-
upe3)]}</ManagedElementOid>
     </scheduler.ITransactionManagerDeviceWrapper>
    </value>
   </param>
   <param name="transactionInputParams">
    <value>
     <scheduler.ITransactionManagerJobInputParameter
type="scheduler.ITransactionManagerJobInputParameter" instance_id="0">
      <TransactionInputParams type="IMObjects_Array">
       <scheduler.ITransactionInputParameters
type="scheduler.ITransactionInputParameters" instance_id="0">
        <ParamName type="String">command</ParamName>
        <ParamValue type="String">show version</ParamValue>
       </scheduler.ITransactionInputParameters>
      </TransactionInputParams>
     </scheduler.ITransactionManagerJobInputParameter>
      </value>
   </param>
   <param name="jobTriggerParams">
    <value>
     <scheduler.IJobTriggerParams type="scheduler.IJobTriggerParams"
instance_id="0">
      <TriggerType type="String">AS_SOON_AS</TriggerType>
      <RunIndefinite type="String">true</RunIndefinite>
      <RepeatInterval type="String">15</RepeatInterval>
      <StartDate type="String">02/25/2013</StartDate>
      <StartTime type="String">xxx</StartTime>
      <DaysofWeek type="String"></DaysofWeek>
      <DaysofMonth type="String"></DaysofMonth>
     </scheduler.IJobTriggerParams>
     </value>
      </param>
         <param name="transactionJobParams">
       <value>
```

```
         <scheduler.ITransactionManagerJobParams
type="scheduler.ITransactionManagerJobParams" instance_id="0">
               <JobName type="String">transaction_manager_job</JobName>
               <Description type="String">this is transaction manager
job</Description>
         </scheduler.ITransactionManagerJobParams>
       </value>
      </param>
  </command>
  .
```

In the above command all devices will refer to generic input params settings defined under 'transactionInputParams' block.

Command Response

```
{[Job(Name=transaction_manager_job-04-22-2013 03:10:36)]}
```

5.10.2.4 Create Transaction Manager Job with device specific input params

The following command example shows the usage of the BQL **CreateTransactionManagerJob** command to create a transaction manager job with device specific input params

```
<command name="CreateTransactionManagerJob">
  <param name="transactionDetails">
    <value>
       <scheduler.ITransactionDetail type="scheduler.ITransactionDetail"
instance_id="4">
        <Description type="String"></Description>
        <PackageID type="String">NewProject</PackageID>
        <TransactionName type="String">executeCommand</TransactionName>
       </scheduler.ITransactionDetail>
     </value>
   </param>
   <param name="devices">
      <value>
        <scheduler.ITransactionManagerDeviceWrapper
type="scheduler.ITransactionManagerDeviceWrapper" instance_id="0">
         <ManagedElementOid type="Oid">{[ManagedElement(Key=c4-
upe1)]}</ManagedElementOid>
          <TransactionInputParams type="IMObjects_Array">
            <scheduler.ITransactionInputParameters
type="scheduler.ITransactionInputParameters" instance_id="0">
             <ParamName type="String">command</ParamName>
            <ParamValue type="String">show version</ParamValue>
             </scheduler.ITransactionInputParameters>
           </TransactionInputParams>
        </scheduler.ITransactionManagerDeviceWrapper>
        <scheduler.ITransactionManagerDeviceWrapper
type="scheduler.ITransactionManagerDeviceWrapper" instance_id="1">
         <ManagedElementOid type="Oid">{[ManagedElement(Key=c4-
upe2)]}</ManagedElementOid>
          <TransactionInputParams type="IMObjects_Array">
```

```
            <scheduler.ITransactionInputParameters
type="scheduler.ITransactionInputParameters" instance_id="0">
             <ParamName type="String">command</ParamName>
            <ParamValue type="String">show clock</ParamValue>
             </scheduler.ITransactionInputParameters>
           </TransactionInputParams>
         </scheduler.ITransactionManagerDeviceWrapper>
         <scheduler.ITransactionManagerDeviceWrapper
type="scheduler.ITransactionManagerDeviceWrapper" instance_id="2">
          <ManagedElementOid type="Oid">{[ManagedElement(Key= c4-
upe3)]}</ManagedElementOid>
           <TransactionInputParams type="IMObjects_Array">
             <scheduler.ITransactionInputParameters
type="scheduler.ITransactionInputParameters" instance_id="0">
              <ParamName type="String">command</ParamName>
            <ParamValue type="String">show configuration</ParamValue>
             </scheduler.ITransactionInputParameters>
           </TransactionInputParams>
         </scheduler.ITransactionManagerDeviceWrapper>
      </value>
     </param>
     <param name="jobTriggerParams">
      <value>
        <scheduler.IJobTriggerParams type="scheduler.IJobTriggerParams"
instance_id="0">
         <TriggerType type="String">AS_SOON_AS</TriggerType>
         <RunIndefinite type="String">true</RunIndefinite>
         <RepeatInterval type="String">15</RepeatInterval>
         <StartDate type="String">02/25/2013</StartDate>
         <StartTime type="String">xxx</StartTime>
         <DaysofWeek type="String"></DaysofWeek>
         <DaysofMonth type="String"></DaysofMonth>
        </scheduler.IJobTriggerParams>
      </value>
     </param>
        <param name="transactionJobParams">
      <value>
        <scheduler.ITransactionManagerJobParams
type="scheduler.ITransactionManagerJobParams" instance_id="0">
             <JobName type="String">transaction_manager_job</JobName>
             <Description type="String">this is transaction manager
job</Description>
        </scheduler.ITransactionManagerJobParams>
      </value>
     </param>
</command>
.
```

In the above command all devices will use specific input params settings defined along with the device in ‘device’ block.

Command Response

```
{[Job(Name=transaction_manager_job-04-22-2013 04:10:36)]}
```

5.10.2.5 Create Transaction Manager Job with all devices and device specific input params

The following command example shows the usage of the BQL **CreateTransactionManagerJob** command to create a transaction manager job with all devices and device specific input params

```
<command name="CreateTransactionManagerJob">
  <param name="transactionDetails">
    <value>
       <scheduler.ITransactionDetail type="scheduler.ITransactionDetail"
instance_id="4">
        <Description type="String"></Description>
        <PackageID type="String">NewProject</PackageID>
        <TransactionName type="String">executeCommand</TransactionName>
       </scheduler.ITransactionDetail>
     </value>
   </param>
   <param name="devices">
      <value>
        <scheduler.ITransactionManagerDeviceWrapper
type="scheduler.ITransactionManagerDeviceWrapper" instance_id="0">
         <ManagedElementOid type="Oid">{[ManagedElement(Key= c4-
upe1)]}</ManagedElementOid>
          <TransactionInputParams type="IMObjects_Array">
            <scheduler.ITransactionInputParameters
type="scheduler.ITransactionInputParameters" instance_id="0">
             <ParamName type="String">command</ParamName>
             <ParamValue type="String">show version</ParamValue>
            </scheduler.ITransactionInputParameters>
          </TransactionInputParams>
        </scheduler.ITransactionManagerDeviceWrapper>

         <scheduler.ITransactionManagerDeviceWrapper
type="scheduler.ITransactionManagerDeviceWrapper" instance_id="1">
         <ManagedElementOid type="Oid">{[ManagedElement(Key= c4-
upe2)]}</ManagedElementOid>
        </scheduler.ITransactionManagerDeviceWrapper>

        <scheduler.ITransactionManagerDeviceWrapper
type="scheduler.ITransactionManagerDeviceWrapper" instance_id="2">
         <ManagedElementOid type="Oid">{[ManagedElement(Key= c4-
upe3)]}</ManagedElementOid>
        </scheduler.ITransactionManagerDeviceWrapper>
     </value>
    </param>
    <param name="transactionInputParams">
      <value>
        <scheduler.ITransactionManagerJobInputParameter
type="scheduler.ITransactionManagerJobInputParameter" instance_id="0">
                 <TransactionInputParams type="IMObjects_Array">
         <scheduler.ITransactionInputParameters
type="scheduler.ITransactionInputParameters" instance_id="0">
              <ParamName type="String">command</ParamName>
              <ParamValue type="String">show clock</ParamValue>
```

```
          </scheduler.ITransactionInputParameters>
       </TransactionInputParams>
       </scheduler.ITransactionManagerJobInputParameter>
     </value>
     </param>
     <param name="jobTriggerParams">
      <value>
        <scheduler.IJobTriggerParams type="scheduler.IJobTriggerParams"
instance_id="0">
                <TriggerType type="String">AS_SOON_AS</TriggerType>
                <RunIndefinite type="String">true</RunIndefinite>
                <RepeatInterval type="String">15</RepeatInterval>
                <StartDate type="String">02/25/2013</StartDate>
                <StartTime type="String">xxx</StartTime>
                <DaysofWeek type="String"></DaysofWeek>
                <DaysofMonth type="String"></DaysofMonth>
        </scheduler.IJobTriggerParams>
      </value>
     </param>
          <param name="transactionJobParams">
      <value>
        <scheduler.ITransactionManagerJobParams
type="scheduler.ITransactionManagerJobParams" instance_id="0">
              <JobName type="String">transaction_manager_job</JobName>
              <Description type="String">this is transaction manager
job</Description>
        </scheduler.ITransactionManagerJobParams>
      </value>
     </param>
</command>
.
```

In the above command device ‘c4-upe1’ will use device specific input params settings defined along with the device in devices block, however devices ‘c4-upe2’ and ‘c4-upe3’ will refer to the all devices input params settings defined under ‘transactionInputParams’ block

Command Response

```
{[Job(Name=transaction_manager_job-04-22-2013 04:10:36)]}
```

5.10.2.6 Reschedule Transaction Manager Job

The following command example shows the usage of the BQL **Reschedule** command to reschedule a transaction manager job.

```
<command name="Reschedule">
  <param name="oid">
    <value>{[Job(Name=transaction_manager_job-04-29-2013
    12:39:05)]}</value>
  </param>
  <param name="jobSpecID">
    <value>41048</value>
  </param>
  <param name="triggerParams">
```

```
    <value>
      <scheduler.IJobTriggerParams type="scheduler.IJobTriggerParams"
    instance_id="0">
        <TriggerType type="String">PERIODICALLY</TriggerType>
        <RunIndefinite type="String">true</RunIndefinite>
        <RepeatInterval type="String">30</RepeatInterval>
        <StartDate type="String">05/03/2013</StartDate>
        <StartTime type="String">17:32</StartTime>
        <DaysofWeek type="String"></DaysofWeek>
        <DaysofMonth type="String"></DaysofMonth>
      </scheduler.IJobTriggerParams>
     </value>
   </param>
</command>
.
```

Deprecated: *This command is depericated from PN Release 4.2.2 as this command only works with 'transaction-manager' job and is not capabale of rescheduling job of any othe type. Instead of this the Generic Reschedule Job Command should be used to reschedule a Job.*

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerCommandResult
type="scheduler.ITransactionManagerCommandResult" instance_id="0">
  <Status type="String">Success</Status>
</scheduler.ITransactionManagerCommandResult>
```

5.10.2.7 Generic Reschedule Job

The following command example shows the usage of the BQL generic **Reschedule** command to reschedule a job of any tye

```
<command name="Reschedule">
 <param name="oid">
  <value>{[Job(Name=showVersion-11-20-2013 11:16:06)]}</value>
 </param>
 <param name="jobType">
  <value>transaction-manager</value>
 </param>
 <param name="triggerParams">
  <value>
   <scheduler.IJobTriggerParams type="scheduler.IJobTriggerParams"
instance_id="0">
    <TriggerType type="String">PERIODICALLY</TriggerType>
    <RunIndefinite type="String">false</RunIndefinite>
    <RepeatInterval type="String">30</RepeatInterval>
    <StartDate type="String">05/03/2013</StartDate>
    <StartTime type="String">17:32</StartTime>
    <DaysofWeek type="String"></DaysofWeek>
    <DaysofMonth type="String"></DaysofMonth>
   </scheduler.IJobTriggerParams>
  </value>
 </param>
.
```

The above command is a generic command and is capable of rescheduling job of any type for eg. In order to reschedule command manager job we need to pass jobType value as 'command-manager'.

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerCommandResult
type="scheduler.ITransactionManagerCommandResult" instance_id="0">
  <Status type="String">Success</Status>
</scheduler.ITransactionManagerCommandResult>
```

5.10.2.8 Get Transaction Manager Job with single filter

The following command example shows the usage of the BQL **GetTransactionManagerJobByFilter** command to fetch all the transaction manager job(s) that matches the single filter criteria passed in the command.

```
<command name="GetTransactionManagerJobByFilter">
 <param name="jobFilters">
   <value>
    <scheduler.IJobFilter type="scheduler.IJobFilter" instance_id="0">
      <FilterCondition type="String">AND</FilterCondition>
      <FilterCriteria type="String">Equals</FilterCriteria>
      <FilterTypeValue type="String">Job SpecId</FilterTypeValue>
      <FilterValue type="String">17018</FilterValue>
    </scheduler.IJobFilter>
   </value>
 </param>
<param name="isCheckMaxLimit">
   <value>true</value>
</param>
</command>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerJob
type="scheduler.ITransactionManagerJob" instance_id="0">
  <Status type="String">Success</Status>
  <TransactionManagerJobDetails type="IMObjects_Array">
    <scheduler.ITransactionManagerJobDetails
type="scheduler.ITransactionManagerJobDetails" instance_id="1">
      <Comments type="String" />
      <JobID type="Long">14022</JobID>
      <JobName type="Oid">{[Job(Name=showVersion-04-19-2013
01:41:43)]}</JobName>
      <JobSpecID type="Long">17018</JobSpecID>
      <JobStatus type="String">Completed</JobStatus>
      <LastRunCompletionTime type="String">Fri Apr 19 2013 13:41:53
IDT</LastRunCompletionTime>
      <LastRunResult type="String">Success</LastRunResult>
```

```
      <LastRunStartTime type="String">Fri Apr 19 2013 13:41:43
IDT</LastRunStartTime>
      <LastRunStatus type="String">Completed</LastRunStatus>
      <NextRunStartTime type="String">N/A</NextRunStartTime>
      <Owner type="String">root</Owner>
    </scheduler.ITransactionManagerJobDetails>
  </TransactionManagerJobDetails>
</scheduler.ITransactionManagerJob>
```

5.10.2.9 Get Transaction Manager Job with multiple filters

The following command example shows the usage of the BQL **GetTransactionManagerJobByFilter** command to fetch all the transaction manager job(s) that matches the multiple filter criteria passed in the command.

```
<command name="GetTransactionManagerJobByFilter">
  <param name="jobFilters">
    <value>
      <scheduler.IJobFilter type="scheduler.IJobFilter"
instance_id="0">
        <FilterCondition type="String">AND</FilterCondition>
        <FilterCriteria type="String">Contains</FilterCriteria>
        <FilterTypeValue type="String">Status</FilterTypeValue>
        <FilterValue type="String">completed</FilterValue>
     </scheduler.IJobFilter>
     <scheduler.IJobFilter type="scheduler.IJobFilter" instance_id="0">
        <FilterCondition type="String">AND</FilterCondition>
        <FilterCriteria type="String">Equals</FilterCriteria>
        <FilterTypeValue type="String">Job SpecId</FilterTypeValue>
        <FilterValue type="String">17018</FilterValue>
     </scheduler.IJobFilter>
   </value>
 </param>
</command>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerJob type="scheduler.ITransactionManagerJob"
instance_id="0">
  <Status type="String">Success</Status>
  <TransactionManagerJobDetails type="IMObjects_Array">
    <scheduler.ITransactionManagerJobDetails
type="scheduler.ITransactionManagerJobDetails" instance_id="1">
      <Comments type="String" />
      <JobID type="Long">14022</JobID>
      <JobName type="Oid">{[Job(Name=showVersion-04-19-2013
01:41:43)]}</JobName>
      <JobSpecID type="Long">17018</JobSpecID>
      <JobStatus type="String">Completed</JobStatus>
      <LastRunCompletionTime type="String">Fri Apr 19 2013 13:41:53
IDT</LastRunCompletionTime>
      <LastRunResult type="String">Success</LastRunResult>
      <LastRunStartTime type="String">Fri Apr 19 2013 13:41:43
IDT</LastRunStartTime>
      <LastRunStatus type="String">Completed</LastRunStatus>
      <NextRunStartTime type="String">N/A</NextRunStartTime>
      <Owner type="String">root</Owner>
    </scheduler.ITransactionManagerJobDetails>
  </TransactionManagerJobDetails>
</scheduler.ITransactionManagerJob>
```

5.10.2.10 Get Transaction Manager Job Result

The following command example shows the usage of the BQL **GetTransactionManagerJobResult** command to retrieve the details of a transaction manager job.

```
<command name="GetTransactionManagerJobResult">
  <param name="oid">
  <value>{[Job(Name=showVersion-04-19-2013 01:41:43)]}</value>
  </param>
</command>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerJobDetails
type="scheduler.ITransactionManagerJobDetails" instance_id="0">
  <Status type="String">Success</Status>
  <TransactionManagerJobTaskDetails type="IMObjects_Array">
    <scheduler.ITransactionManagerJobTaskDetails
type="scheduler.ITransactionManagerJobTaskDetails" instance_id="1">
      <DeviceName type="Oid">{[ManagedElement(Key=N7k)]}</DeviceName>
      <TaskID type="Long">1</TaskID>
      <TaskStatus type="String">SUCCESS</TaskStatus>
      <TransactionDetails type="scheduler.ITransactionDetail"
instance_id="2">
        <PackageID type="String">Demo</PackageID>
        <TransactionName type="String">showVersion</TransactionName>
      </TransactionDetails>
    </scheduler.ITransactionManagerJobTaskDetails>
  </TransactionManagerJobTaskDetails>
</scheduler.ITransactionManagerJobDetails
```

5.10.2.11 Get Transaction Manager Task Result

The following command example shows the usage of the BQL **GetTransactionManagerTaskResult** to retrieve the result of a specific transaction manager task.

```
<command name="GetTransactionManagerTaskResult">
  <param name="taskID">
  <value>1</value>
  </param>
 </command>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerJobTaskResult
type="scheduler.ITransactionManagerJobTaskResult" instance_id="0">
  <DebugTrace type="String">&lt;?xml version="1.0" encoding="UTF-
8"?&gt;&lt;observations/&gt;</DebugTrace>
```

```
  <Result type="String">Error while running XDE procedure
!Device_ShowRunningConfig: Error Message:
&lt;palError&gt;&lt;deviceId&gt;CAT_39&lt;/deviceId&gt;&lt;code&gt;HANDLER_ERROR
&lt;/code&gt;&lt;message&gt;Error while trying to run handler. Action :
[Set_of_Tests4]pal.xpa/executeCBScript/executeCBScript.par, . Error :
com.cisco.nm.cmp.client.common.nbi.SystemException&lt;/message&gt;&lt;handlerCod
e&gt;ERROR_HANDLER_ERROR&lt;/handlerCode&gt;&lt;/palError&gt;</Result>
  <Status type="String">Success</Status>
  <TaskID type="Long">9</TaskID>
  <TransactionLog type="">Null</TransactionLog>
</scheduler.ITransactionManagerJobTaskResult>
```

5.10.2.12 Delete Transaction Manager Job with Single Oid

The following command example shows the usage of the BQL **DeleteJob** command to delete a single transaction manager job.

```
<command name="DeleteJob">
  <param name="oid">
    <value>{[Job(Name=showVersion-04-19-2013 01:42:41)]}</value>
  </param>
  <param name="jobType">
    <value>transaction-manager</value>
   </param>
</command>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerResult
type="scheduler.ITransactionManagerResult" instance_id="0">
  <TransactionManagerCommandResult type="IMObjects_Array">
    <scheduler.ITransactionManagerCommandResult
type="scheduler.ITransactionManagerCommandResult" instance_id="1">
      <JobName type="Oid">{[Job(Name=showVersion-04-19-2013
01:42:41)]}</JobName>
      <Status type="String">Success</Status>
    </scheduler.ITransactionManagerCommandResult>
  </TransactionManagerCommandResult>
</scheduler.ITransactionManagerResult>
```

5.10.2.13 Delete Transaction Manager Job with Multiple Oids

The following command example shows the usage of the BQL **DeleteJob** command to delete multiple transaction manager jobs.

```
<command name="DeleteJob">
  <param name="oids">
    <value>{[Job(Name=transaction_manager_job-03-22-2013
10:47:28)]}</value>
    <value>{[Job(Name=transaction_manager_job-03-21-2013
05:01:42)]}</value>
  </param>
  <param name="jobType">
```

```
      <value>transaction-manager</value>
    </param>
</command>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerResult
type="scheduler.ITransactionManagerResult" instance_id="0">
  <TransactionManagerCommandResult type="IMObjects_Array">
    <scheduler.ITransactionManagerCommandResult
type="scheduler.ITransactionManagerCommandResult" instance_id="1">
      <JobName type="Oid">{[Job(Name=transaction_manager_job-03-15-2013
09:08:43)]}</JobName>
      <Status type="String">Success</Status>
    </scheduler.ITransactionManagerCommandResult>
    <scheduler.ITransactionManagerCommandResult
type="scheduler.ITransactionManagerCommandResult" instance_id="2">
      <JobName type="Oid">{[Job(Name=transaction_manager_job-03-15-2013
09:05:19)]}</JobName>
      <Status type="String">Success</Status>
    </scheduler.ITransactionManagerCommandResult>
  </TransactionManagerCommandResult>
</scheduler.ITransactionManagerResult>
```

5.10.2.14 Update Job Command to Suspend Transaction Manager Job

The following command example shows the usage of the BQL **UpdateJob** command to suspend a transaction manager job.

```
<command name="UpdateJob">
<param name="oid">
  <value>{[Job(Name=showVersion-04-19-2013 01:42:41)]}</value>
</param>
<param name="operationType">
  <value>2</value>
</param>
<param name="jobType">
  <value>transaction-manager</value>
</param>
</command>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerResult
type="scheduler.ITransactionManagerResult" instance_id="0">
  <TransactionManagerCommandResult type="IMObjects_Array">
    <scheduler.ITransactionManagerCommandResult
type="scheduler.ITransactionManagerCommandResult" instance_id="1">
      <JobName type="Oid">{[Job(Name=showVersion-04-19-2013
01:42:41)]}</JobName>
      <Status type="String">Success</Status>
    </scheduler.ITransactionManagerCommandResult>
  </TransactionManagerCommandResult>
```

```
</scheduler.ITransactionManagerResult>
```

5.10.2.15 Update Job Command to Resume Transaction Manager Job

The following command example shows the usage of the BQL **UpdateJob** command to resume a suspended transaction manager job.

```
<command name="UpdateJob">
<param name="oid">
  <value>{[Job(Name=showVersion-04-19-2013 01:42:41)]}</value>
</param>
<param name="operationType">
  <value>0</value>
</param>
<param name="jobType">
  <value>transaction-manager</value>
</param>
</command>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerResult
type="scheduler.ITransactionManagerResult" instance_id="0">
  <TransactionManagerCommandResult type="IMObjects_Array">
    <scheduler.ITransactionManagerCommandResult
type="scheduler.ITransactionManagerCommandResult" instance_id="1">
      <JobName type="Oid">{[Job(Name=showVersion-04-19-2013
01:42:41)]}</JobName>
      <Status type="String">Success</Status>
    </scheduler.ITransactionManagerCommandResult>
  </TransactionManagerCommandResult>
</scheduler.ITransactionManagerResult>
```

5.10.2.16 Update Job Command to cancel a single Transaction Manager Job

The following command example shows the usage of the BQL **UpdateJob** command to cancel a single transaction manager job.

```
<command name="UpdateJob">
  <param name="oid">
     <value>{[Job(Name=showVersion-04-19-2013 01:42:41)]}</value>
   </param>
   <param name="operationType">
      <value>3</value>
   </param>
   <param name="jobType">
       <value>transaction-manager</value>
   </param>
</command>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerResult
type="scheduler.ITransactionManagerResult" instance_id="0">
```

```
  <TransactionManagerCommandResult type="IMObjects_Array">
    <scheduler.ITransactionManagerCommandResult
type="scheduler.ITransactionManagerCommandResult" instance_id="1">
      <JobName type="Oid">{[Job(Name=showVersion-04-19-2013
01:42:41)]}</JobName>
      <Status type="String">Success</Status>
    </scheduler.ITransactionManagerCommandResult>
  </TransactionManagerCommandResult>
</scheduler.ITransactionManagerResult>
```

5.10.2.17 Update Job Command to cancel multiple Transaction Manager Jobs

The following command example shows the usage of the BQL **UpdateJob** command to cancel multiple transaction manager jobs.

```
<command name="UpdateJob">
  <param name="oids">
    <value>{[Job(Name=transaction_manager_job-04-03-2013
02:11:39)]}</value>
    <value>{[Job(Name=transaction_manager_job-03-15-2013
09:33:23)]}</value>
  </param>
  <param name="operationType">
    <value>3</value>
  </param>
  <param name="jobType">
    <value>transaction-manager</value>
  </param>
</command>
.
```

Command Response

```
<?xml version="1.0" encoding="UTF-8"?>
<scheduler.ITransactionManagerResult type="scheduler.ITransactionManagerResult"
instance_id="0">
 <TransactionManagerCommandResult type="IMObjects_Array">
   <scheduler.ITransactionManagerCommandResult
type="scheduler.ITransactionManagerCommandResult" instance_id="1">
    <JobName type="Oid">{[Job(Name=transaction_manager_job-04-03-2013
02:11:39)]}</JobName>
    <Status type="String">Success</Status>
  </scheduler.ITransactionManagerCommandResult>
  <scheduler.ITransactionManagerCommandResult
type="scheduler.ITransactionManagerCommandResult" instance_id="2">
     <JobName type="Oid">{[Job(Name=transaction_manager_job-03-15-2013
09:33:23)]}</JobName>
     <Status type="String">Success</Status>
   </scheduler.ITransactionManagerCommandResult>
 </TransactionManagerCommandResult>
</scheduler.ITransactionManagerResult>
```

5.11 BQL Error Catalog and Examples

- This topic includes the following sections:BQL Error Handling, page 236

5.11.1 BQL Error Handling

This section describes the BQL error message format, which presents the error message as ISystemError IMO in XML format.

ISystemError contains an identifier field that has an error code, a string description field, and a string array field named ErrorStackTrace containing an exception stack trace, if present, as shown in the following example:

```
<ISystemError type=”ISystemError” instance_id=”1>
  <ID type="Oid">{[SystemError(Code=ERROR CODE GOES HERE)]}</ID>
  <Description type="String">DESCRIPTION OF THE ERROR</Description>
  <ErrorStackTrace type="java.lang.String_Array">
        <java.lang.String>StackEntry 1</java.lang.String>
        <java.lang.String>StackEntry 2</java.lang.String>
        ...
        <java.lang.String>StackEntry N</java.lang.String>
  </ErrorStackTrace>
</ISystemError>
```

The following sections display example error messages in XML format.

BQL Error: Example 1

```
<?xml version="1.0" encoding="UTF-8"?>
<ISystemError type=”ISystemError” instance_id=”1>
  <ID type="Oid">{[SystemError(Code=1000)]}</ID>
  <Description type="String">ERROR (1000): General error, Exception:
  java.lang.IllegalArgumentException: Template Simple.template not
  found</Description>
  <ErrorStackTrace type="java.lang.String_Array">

  <java.lang.String>com.sheer.system.os.services.workflow.dwe.InternalWorkf
  lowUtil.getTempla teIdByName(InternalWorkflowUtil.java:80)
  </java.lang.String>

  <java.lang.String>com.sheer.system.os.services.workflow.dwe.InternalWorkf
  lowUtil.getTempla teIdByName(InternalWorkflowUtil.java:42)
  </java.lang.String>
    <java.lang.String>com.sheer.system.os.services.workflow.WorkflowService
    Impl.runWorkflo w(WorkflowServiceImpl.java:373)</java.lang.String>
    <java.lang.String>sun.reflect.NativeMethodAccessorImpl.invoke0(Native
    Method)</java.lang.String>
    <java.lang.String>sun.reflect.NativeMethodAccessorImpl.invoke(NativeM
    ethodAccessorImpl
    .java:39)
  </java.lang.String>
    <java.lang.String>sun.reflect.DelegatingMethodAccessorImpl.invoke(Dele
    gatingMethodAcce ssorImpl.java:25)</java.lang.String>
    <java.lang.String>java.lang.reflect.Method.invoke(Method.java:324)</j
    ava.lang.String>
    <java.lang.String>com.sheer.system.os.services.base.BaseOSService.inv
    oke(BaseOSService
    .java:57)>
  </java.lang.String>
```

```
        <java.lang.String>com.sheer.system.os.services.management.ServiceW
        rapper.executeComman d(ServiceWrapper.java:202)</java.lang.String>
        <java.lang.String>com.sheer.system.os.services.management.Manageme
        ntService.execute(Ma nagementService.java:208)</java.lang.String>

<java.lang.String>com.sheer.system.os.services.management.adapters.rtc.Remo
teTransportConn
ectorAgent.executeLocal(RemoteTransportConnectorAgent.java:157)</java.lang
.String>

<java.lang.String>com.sheer.system.os.services.management.adapters.rtc.Remo
teTransportConn
ectorAgent.handleExecuteMessage(RemoteTransportConnectorAgent.java:102)</ja
va.lang.String>
<java.lang.String>com.sheer.system.os.services.management.adapters.rtc.Rem
oteTransportConn
ectorAgent.processMessage(RemoteTransportConnectorAgent.java:66)</java.lan
g.String>
  <java.lang.String>com.sheer.system.agentshell.AgentBase.run(AgentBase.ja
  va:232)</java. lang.String>
  <java.lang.String>com.sheer.system.os.services.scheduler.OSAgent.run(OSA
  gent.java:107)
>
</java.lang.String>
  <java.lang.String>com.sheer.system.os.util.ThreadPool$OSThread.run(Threa
  dPool.java:272
)>
</java.lang.String>
  </ErrorStackTrace>
</ISystemError>
```

BQL Error: Example 2

```
  <?xml version="1.0" encoding="UTF-8"?>
  <ISystemError type=”ISystemError” instance_id=”1>
        <ID type="Oid">{[SystemError(Code=1200)]}</ID>
        <Description type="String">Invalid Command Syntax.
  java.lang.IllegalArgumentException: Invalid command syntax. no such
  command</Description>
  </ISystemError>
```

BQL Error: Example 3

```
  <?xml version="1.0" encoding="UTF-8"?>
  <ISystemError type=”ISystemError” instance_id=”1>
        <ID type="Oid">{[SystemError(Code=1200)]}</ID>
        <Description type="String">Error parsing XML.
  org.jdom.input.JDOMParseException: Error on line 1: The element type
  "asdf" must be terminated by the matching end-tag
  "&lt;/asdf&gt;".</Description>
  </ISystemError>
```

BQL Error: Example 4

```
  <?xml version="1.0" encoding="UTF-8"?>
  <ISystemError type=”ISystemError” instance_id=”1>
        <ID type="Oid">{[SystemError(Code=1202)]}</ID>
```

```
        <Description type="String">Error executing command
java.lang.Exception: ShellRaw: null cid returned (Unknown
command)</Description>
</ISystemError>
```

The old error message format is supported for backward compatibility. The default system error message format is configurable in the registry in mmvm/agents/adapters/ShellRawServer/XMLErrorReporting. By default, the value is true, which indicates that the BQL reports errors are in the new XML error message format.

Note Changes to the registry should be performed only with the support of Cisco. For details, contact your Cisco account representative.

You can switch between two modes during each BQL session:

1. To enable the old non-XML error message format, enter **`xmlerrorreporting off`**.
   The server responds with the following:
   XML Error Reporting was set to OFF.
2. To enable the new XML error message format, enter **`xmlerrorreporting on`**.
   The server responds with the following:
   XML Error Reporting was set to ON.

To check the current XML error message format, enter **`xmlerrorreporting`**. The server response is one of the following:

- XML Error Reporting is : ON
- XML Error Reporting is : OFF

5.11.2 General BQL Errors

Table 5-18 lists the BQL errors that can be encountered frequently:

Table 5-18 BQL Errors

Error Message	Possible Cause	Possible Action
Command got timeout.	VNE is either not accessible or in maintenance mode.	Restart VNE.
Invalid command name.	The command is not loaded on the VNE –or— The command name is wrong in the BQL request file.	Check if the command is loaded on the corresponding VNE. –or— Check the command name in the BQL request file. The command name should be same as the command that is loaded on the VNE.

Error Message	Possible Cause	Possible Action
Invalid command syntax. No such command.	In the BQL request file, some of the parameters are either misspelt or is missing.	Check all parameters in the BQL request.
Unrecognized command	—	—
Communication/protocol failure.	The device in unreachable in Prime Network.	Get the device in reachable state.

In addition to the errors listed in the Table 5-18, you may also encounter the following errors:

- Garbage Characters Input, page 239
- Invalid XML Format, page 239
- Invalid Command Name, page 240
- Invalid Parameter Name, page 240
- Invalid OID Value, page 240
- VNE Does Not Exist (Get Command) , page 241

Garbage Characters Input

Short Description
Garbage characters input

Long Description
Enter garbage characters

Error Example
Invalid Command Syntax:

```
java.lang.IllegalArgumentException: Invalid command syntax. no such
command
```

Error Code
None

Invalid XML Format

Short Description
Invalid XML format

Long Description
This is an invalid XML format

Error Example
Error parsing XML:

```
org.jdom.input.JDOMParseException: Error on line 1: Element type "key"
must be followed by either attribute specifications, ">" or "/>".
```

Error Code
None

Invalid Command Name

Short Description
Invalid command name

Long Description
This is an invalid command name

Error Example
Error executing command:
```
java.lang.Exception: ShellRaw: Unknown command(null cid returned)
```

Error Code
None

Invalid Parameter Name

Short Description
Invalid parameter name

Long Description
This is an invalid parameter name

Error Example
Error executing command:
```
java.lang.Exception: ShellRaw: Unknown command(null cid returned)
```

Error Code
None

Invalid OID Value

Short Description
Invalid OID value

Long Description
This is an invalid OID value

Error Example
Invalid Command Syntax:
```
java.lang.IllegalArgumentException: Invalid command syntax. Argument
"oid" is invalid.
```

Error Code
None

VNE Does Not Exist (Get Command)

Short Description
VNE does not exist (**Get** command)

Long Description
This VNE does not exist (**Get** command)

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:2d [64]13Destination: (CL.TS-64.103.124.248 [2]-
0:0:0:0:0:0:0:8 [64]13Exception: ERROR (5103): Agent doesn't exist
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.getXIDFro mKey(ElementManagementElementHandler.java:499)
at
com.sheer.metromission.plugin.snapshot.SnapshotElementGetCommand$FirstSt
ateHandler.sendRequestToMaps(SnapshotElementGetCommand.java:240)
at
com.sheer.metromission.plugin.snapshot.SnapshotElementGetCommand$FirstSt
ateHandler.handle( SnapshotElementGetCommand.java:142)
at
com.sheer.framework.commands.MultiStateCommand.localExecute(MultiStateCo
mmand.java:59)at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
------------------------------------------------------------------
```

Error Code
5103

5.11.3 Command Builder Command BQL Errors

This section includes the following errors:

Script Name Is Blank

Short Description
Script name is blank

Long Description
The script name is blank

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:2f [64]13Destination: (CL.TS-64.103.124.248 [2]-
0:0:0:0:0:0:0:9 [64]13Exception: java.lang.IllegalArgumentException:
Error! Script does not have a name.
at
com.sheer.metrocentral.coretech.common.maps.command.PublishUtils.validate
PublishableElemen tOid(PublishUtils.java:509)
at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j ava:253)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
```

```
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code
None

Script Name Contains Invalid Characters

Short Description
Script name contains invalid characters

Long Description
The script name contains invalid characters

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:30 [64]13Destination: (CL.TS-64.103.124.248 [2]-
0:0:0:0:0:0:0:a [64]13Exception: java.lang.IllegalArgumentException:
Error! Script name can contain only alphanumeric characters,"-" & "_",
invalid character at pos 5
at
com.sheer.metrocentral.coretech.common.maps.command.PublishUtils.validat
ePublishableElemen tOid(PublishUtils.java:515)
at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j ava:253)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code
None

Registry Path Is Blank

Short Description

Registry path is blank

Long Description
The registry path is blank

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:31 [64]13Destination: (CL.TS-64.103.124.248 [2]-
0:0:0:0:0:0:0:b [64]13Exception: java.lang.IllegalArgumentException:
Error! Script registry path cannot be null/blank. at
com.sheer.metrocentral.coretech.common.maps.command.PublishUtils.validat
ePublishableElementOid(PublishUtils.java:512)
at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j ava:253)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code
None

Context IMO Is Null

Short Description
Context IMO is null

Long Description
The context IMO is null

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:32
[64]13Destination: (CL.TS-64.103.124.248 [2]-0:0:0:0:0:0:0:c
[64]13Exception: java.lang.IllegalArgumentException: Error! Script does
not have a context IMO. at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j
```

```
ava:266)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

None

Context IMO Class Is Invalid

Short Description

Context IMO class is invalid

Long Description

The context IMO class is invalid

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:33 [64]13Destination: (CL.TS-64.103.124.248 [2]-
0:0:0:0:0:0:0:d [64]13Exception: java.lang.IllegalArgumentException:
Error! The Context IMO Type "com.sheer.imo.IManagedElemen" is invalid!
at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j ava:271)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code

None

Language Type Is Unknown

Short Description
Language type is unknown

Long Description
The language type entered is unknown

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:34 [64]13Destination: (CL.TS-64.103.124.248 [2]-
0:0:0:0:0:0:0:e [64]13Exception: java.lang.IllegalArgumentException:
Unknown language type 50! Only Sheer1(0) & Binshell(1) languages are
currently supported.
at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j ava:286)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code
None

Protocol Type Is Unknown

Short Description
Protocol type is unknown

Long Description
The protocol type is unknown

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
```

```
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:35
[64]13Destination: (CL.TS-64.103.124.248 [2]-0:0:0:0:0:0:0:f
[64]13Exception: java.lang.IllegalArgumentException: Unknown protocol
type 50! Only Telnet(0) & SNMP(1) protocols are currently supported.
at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDef
aults(SetScript.j ava:289)at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code

None

Sheer1 and SNMP Protocol

Short Description

Sheer1 and SNMP Protocol

Long Description

Sheer1 and SNMP Protocol

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:36
[64]13Destination: (CL.TS-64.103.124.248 [2]-0:0:0:0:0:0:0:10
[64]13Exception: java.lang.IllegalArgumentException: Error! Script
Language "Sheer1" supports only the Telnet protocol.
at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j ava:292)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
```

```
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
None

Missing Roles

Short Description
Missing roles

Long Description
There are missing roles

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:37
[64]13Destination: (CL.TS-64.103.124.248 [2]-0:0:0:0:0:0:0:11
[64]13Exception: java.lang.IllegalArgumentException: Error! Script must
have at least one role. at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDef
aults(SetScript.j
ava:295)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.jav
a:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComma
nd.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code
None

Invalid Role Entered

Short Description
Invalid role entered

Long Description
Invalid role entered

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:38 [64]13Destination: (CL.TS-64.103.124.248 [2]-
0:0:0:0:0:0:0:12 [64]13Exception: java.lang.IllegalArgumentException:
Error! Script role Administrator does not exist. at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j
ava:306)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

None

Time of the Script Is Too Short

Short Description

Time of the script is too short

Long Description

The time period defined for the script is too short

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:39
[64]13Destination: (CL.TS-64.103.124.248 [2]-0:0:0:0:0:0:0:13
[64]13Exception: java.lang.IllegalArgumentException: Error! Script
timeout must be over 1000ms. at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j
ava:311)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
```

```
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code
None.

Script Argument Name Is Blank

Short Description
Script argument name is blank

Long Description
The script argument name is blank

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:3a [64]13Destination: (CL.TS-64.103.124.248 [2]-
0:0:0:0:0:0:0:14 [64]13Exception: java.lang.IllegalArgumentException:
Error! one of the script parameters name is null. at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j
ava:319)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code
None

Script Argument Type Is Blank

Short Description
Script argument type is blank

Long Description
The script argument type is blank

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:3b
[64]13Destination: (CL.TS-64.103.124.248 [2]-0:0:0:0:0:0:0:15
[64]13Exception: java.lang.IllegalArgumentException: Error! Script
parameter aaa type is empty. at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j
ava:326)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code
None

Script Argument Type Is Invalid

Short Description
Script argument type is invalid

Long Description
The script argument type is invalid

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:3c [64]13Destination: (CL.TS-64.103.124.248 [2]-
0:0:0:0:0:0:0:16 [64]13Exception: java.lang.IllegalArgumentException:
Error! Script parameter aaa type is invalid at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDe
faults(SetScript.j
ava:332)
```

```
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

None

Script Enumeration Is Invalid

Short Description

Script enumeration is invalid

Long Description

The script enumeration is invalid

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813258]-
0:0:0:0:0:0:0:3d
[64]13Destination: (CL.TS-64.103.124.248 [2]-0:0:0:0:0:0:0:17
[64]13Exception:
java.lang.Exception: Error in enums for argument 4343=555;3434 at
com.sheer.client.common.components.cmdbld.CommandManagerUtils.parseAgent
Enums(CommandManag erUtils.java:861)
at
com.sheer.metromission.plugin.client.commands.SetScript.validateAndSetDef
aults(SetScript.j ava:343)
at
com.sheer.metromission.plugin.client.commands.SetScript.run(SetScript.ja
va:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code
None

Script Does Not Exist

Short Description
Script does not exist

Long Description
The script does not exist

Error Example
```
Error executing command:
java.lang.Exception: ShellRaw: Unknown command(null cid returned)
```

Error Code
None

VNE Was Loaded Once But Now It Is Down

Short Description
VNE was loaded once but now it is down

Long Description
The VNE was loaded once but now it is down

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813259]-
0:0:0:0:0:0:0:0 [64]13Destination: (CL.TS-64.103.124.248 [3]-
0:0:0:0:0:0:0:4 [64]13Exception: ERROR (2302): VNE Is Not Loaded. -
java.lang.reflect.InvocationTargetExceptionat
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.jav
a:39)
at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessor
Impl.java:25)
at java.lang.reflect.Method.invoke(Method.java:324) at
com.sheer.metromission.commandmanager.CommandManagerImpl.createCommand(C
ommandManagerImpl. java:473)
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Ses
sionCommandContain er.java:155)
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Ses
sionCommandContain er.java:163)
```

```
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Ses
sionCommandContain er.java:142)
at
com.sheer.metromission.session.SessionCommandContainer.createCommandEntr
y(SessionCommandCo ntainer.java:191)
at
com.sheer.metromission.session.Session.processMessage(Session.java:307)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
Caused by: java.lang.IllegalArgumentException: Failed to create dynamic
proxy: ERROR (5118): VNE Not Loaded.
at
com.sheer.framework.commands.MCDefaultCommand.createAgentId(MCDefaultCom
mand.java:170)
at
com.sheer.framework.commands.MCDefaultCommand.setParam(MCDefaultCommand.
java:61)
... 13 more at
com.sheer.metromission.commandmanager.CommandManagerImplErrorHandler.cre
ateMetroMissionCom mandException(CommandManagerImplErrorHandler.java:62)
at
com.sheer.metromission.commandmanager.CommandManagerImplErrorHandler.sig
nifyException(Comm andManagerImplErrorHandler.java:38)
at
com.sheer.metromission.commandmanager.CommandManagerImpl.createCommand(C
ommandManagerImpl. java:487)
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Ses
sionCommandContain er.java:155)
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Ses
sionCommandContain er.java:163)
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Ses
sionCommandContain er.java:142)
at
com.sheer.metromission.session.SessionCommandContainer.createCommandEntr
y(SessionCommandCo ntainer.java:191)
at
com.sheer.metromission.session.Session.processMessage(Session.java:307)
```

Error Code

2302

VNE Was Loaded Once But Now It Has Been Deleted

Short Description

VNE was loaded once but now it has been deleted

Long Description

The VNE was loaded once but now it has been deleted

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813259]-
0:0:0:0:0:0:0:0
[64]13Destination: (CL.TS-64.103.124.248 [3]-0:0:0:0:0:0:0:6
[64]13Exception: ERROR (2301): VNE Does Not Exist. -
java.lang.reflect.InvocationTargetExceptionat
sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.jav
a:39)
at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessor
Impl.java:25)
at java.lang.reflect.Method.invoke(Method.java:324) at
com.sheer.metromission.commandmanager.CommandManagerImpl.createCommand(C
ommandManagerImpl. java:473)
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Sess
ionCommandContain er.java:155)
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Ses
sionCommandContain er.java:163)
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Ses
sionCommandContain er.java:142)
at
com.sheer.metromission.session.SessionCommandContainer.createCommandEntr
y(SessionCommandCo ntainer.java:191)
at
com.sheer.metromission.session.Session.processMessage(Session.java:307)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
Caused by: java.lang.IllegalArgumentException: Failed to create dynamic
proxy: ERROR (5103): Agent doesn't exist
at
com.sheer.framework.commands.MCDefaultCommand.createAgentId(MCDefaultCom
mand.java:170)
at
com.sheer.framework.commands.MCDefaultCommand.setParam(MCDefaultCommand.
java:61)
... 13 more at
com.sheer.metromission.commandmanager.CommandManagerImplErrorHandler.cre
ateMetroMissionCom mandException(CommandManagerImplErrorHandler.java:62)
at
com.sheer.metromission.commandmanager.CommandManagerImplErrorHandler.sig
nifyException(Comm andManagerImplErrorHandler.java:43)
at
com.sheer.metromission.commandmanager.CommandManagerImpl.createCommand(Co
mmandManagerImpl. java:487)
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Sess
ionCommandContain er.java:155)
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Ses
sionCommandContain er.java:163)
```

```
at
com.sheer.metromission.session.SessionCommandContainer.createCommand(Ses
sionCommandContain er.java:142)
at
com.sheer.metromission.session.SessionCommandContainer.createCommandEntr
y(SessionCommandCo ntainer.java:191)
at
com.sheer.metromission.session.Session.processMessage(Session.java:307)
```

Error Code

2301

Timeout for One of the Scripts Lines Has Expired

Short Description

Timeout for one of the scripts lines has expired

Long Description

The timeout for one of the script's lines has expired

Error Example

```
<?xml version="1.0" encoding="UTF-8"?>
<IScriptResult>
<ID
type="Oid">{[ScriptResult(ScriptName=test)(Sequence=1166451578843)]}</I
D>
<ExecutionSequence type="IMObjects_Array">
<IScriptEvent>
<ID type="Oid">{[ScriptEvent(Index=1)]}</ID>
<EventTypeEnum type="Integer">1</EventTypeEnum>
<Message type="String">PE-West#ping 44.44.44.44</Message>
</IScriptEvent>
<IScriptEvent>
<ID type="Oid">{[ScriptEvent(Index=2)]}</ID>
<EventTypeEnum type="Integer">2</EventTypeEnum>
<Message type="String">PE-West#</Message>
</IScriptEvent>
<IScriptEvent>
<ID type="Oid">{[ScriptEvent(Index=3)]}</ID>
<EventTypeEnum type="Integer">6</EventTypeEnum>
<Message type="String">receiveUntil(): general timeout
expired(value=1000)(ping 44.44.44.44

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 44.44.44.44, timeout is 2 seconds:
)</Message>
</IScriptEvent>
</ExecutionSequence>
<ExecutionTime type="Long">1312</ExecutionTime>
<FailedActivity type="String" />
<StatusEnum type="Integer">2</StatusEnum>
</IScriptResult>
```

Error Code

None

The Script's Total Timeout Has Expired

Short Description
The script's total timeout has expired

Long Description
The script's total timeout has expired

Error Example
```
Command got timeout------
com.sheer.framework.commands.messages.TimeoutMessage ------
com.sheer.framework.commands.messages.TimeoutMessage, source=(MM.SA-
64.103.124.248 [6813261]-0:0:0:0:0:0:0:54 [64], destination=(CL.TS-
64.103.124.248 [5]-0:0:0:0:0:0:0:4 [64], id=0
---------------------------------------------------------------
```

Error Code
None

Success Pragma Failure

Short Description
Success pragma failure

Long Description
Success pragma failure

Error Example
```
<"Integer">6</EventTypeEnum>
<Message type="String"> ^ Failed to find the text "success" in the
device reply!, script terminated.</Message>
</IScriptEvent>
</ExecutionSequence>
<ExecutionTime type="Long">10078</ExecutionTime>
<FailedActivity type="String" />
<StatusEnum type="Integer">2</StatusEnum>
</IScriptResult>
```

Error Code
None

Fail Pragma Failure

Short Description
Fail pragma failure

Long Description

Fail pragma failure

Error Example

```
<"Integer">6</EventTypeEnum>
<Message type="String"> ^ Found the text "0 percent" in the device
reply!, script terminated.</Message>
</IScriptEvent>
</ExecutionSequence>
<ExecutionTime type="Long">10078</ExecutionTime>
<FailedActivity type="String" />
<StatusEnum type="Integer">2</StatusEnum>
</IScriptResult>
```

Error Code

None

5.11.4 Inventory BQL Errors

This section includes the error details for VNE OID Does Not Exist:

Short Description

VNE OID does not exist

Long Description

The VNE OID does not exist

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813259]-
0:0:0:0:0:0:0:4b
[64]13Destination: (CL.TS-64.103.124.248 [3]-0:0:0:0:0:0:0:8
[64]13Exception:
java.lang.Exception: oid
{[ManagedElement(Key=PE-
West)][PhysicalRoot][Chassis][Slot(SlotNum=1)][Module][Port(PortNumber=Fa
stEthernet1/2)]} does not exist
at com.sheer.metrocentral.framework.maps.command.Get.get(Get.java:219)
at com.sheer.metrocentral.framework.maps.command.Get.get(Get.java:236)
at
com.sheer.metrocentral.framework.maps.command.Get$StartStateHandler.hand
le(Get.java:571) at
com.sheer.framework.commands.MultiStateCommand.localExecute(MultiStateCo
mmand.java:59) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.system.agentshell.components.cre.CommandRunEnvironment.handleE
xecuteMessage(Comm andRunEnvironment.java:305)
at
com.sheer.system.agentshell.components.cre.CommandRunEnvironment$Execute
MessageHandler.handle(CommandRunEnvironment.java:433)
at
com.sheer.system.agentshell.components.cre.CommandRunEnvironment.process
Message(CommandRunEnvironment.java:183)
at com.sheer.metrocentral.framework.da.DA.processMessage(DA.java:319)
```

```
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)

------------------------------------------------------------------------
```

Error Code
None

5.11.5 Cisco Prime Network Administration BQL Errors

This section includes the following errors:

Creating an AVM for a Unit That Does Not Exist

Short Description

Creating an AVM for a unit that does not exist

Long Description

Trying to create an AVM for a unit that does not exist

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:5a
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:a
[64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.valid
ateElementOid(Uni tElementHandler.java:965)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification
(BOSManageCommandUtil.java:49) at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateAvm.run(Create
Avm.java:113) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
```

```
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
-----------------------------------------------------------------
```

Error Code

7001

Creating an AVM with a Reserved AVM ID

Short Description

Creating an AVM with a reserved AVM ID

Long Description

Trying to create an AVM with a reserved AVM ID

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:5c [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:b [64]13Exception: ERROR (5113): This is a reserved AVM
number. Action not allowed on reserved AVMs.
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valida
teNewAvm(AvmEleme ntHandler.java:166)
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.addEl
ement(AvmElementHa ndler.java:89)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElmentH
andler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.b
eforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.add(DataContainer.java:154) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(
ImoNotificationUti l.java:293)
```

```
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateAvm.run(CreateA
vm.java:113) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code

5113

Deleting an AVM from a Unit That Does Not Exist

Short Description

Deleting an AVM from a unit that does not exist

Long Description

Trying to delete an AVM from a unit that does not exist

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
```

```
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:5e [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:c [64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.vali
dateElementOid(Uni tElementHandler.java:965)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteAvm.deleteAvm(
DeleteAvm.java:155
)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteAvm.run(Delete
Avm.java:148) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
---------------------------------------------------------------------
```

Error Code

7001

Deleting a Reserved AVM

Short Description

Deleting a reserved AVM

Long Description

Trying to delete a reserved AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:60
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:d
[64]13Exception: ERROR (5113): This is a reserved AVM number. Action not
allowed on reserved AVMs.
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valida
teDeleteAvm(AvmEl ementHandler.java:261)
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.remove
Element(AvmElemen tHandler.java:281)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElmentH
andler.updateElem ent(AbstractBosManageElmentHandler.java:87)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.remove(DataContainer.java:244)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeFromIMO
(ImoNotificationUt il.java:313)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeIMOsfro
mIMO(ImoNotificati onUtil.java:307)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:162)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
```

```
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteAvm.deleteAvm(
DeleteAvm.java:155)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteAvm.run(Delet
eAvm.java:148)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code

5113

Deleting a Nonexistent AVM

Short Description

Deleting a nonexistent AVM

Long Description

Trying to delete a nonexistent AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:64 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:f [64]13Exception: ERROR (5109): AVM does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE
lementHandler.java:932) at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateDeleteAvm(AvmEl ementHandler.java:258)
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.remov
eElement(AvmElemen tHandler.java:281)
```

```
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:87)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoChangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.remove(DataContainer.java:244)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeFromIMO
(ImoNotificationUt il.java:313)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeIMOsfro
mIMO(ImoNotificati onUtil.java:307)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:162)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessCo
mmandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificatio
nUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteAvm.deleteAvm(
DeleteAvm.java:155
)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteAvm.run(DeleteA
vm.java:148) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
```

```
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code
5109

Creating an AVM with an ID That Already Exists

Short Description
Creating an AVM with an ID that already exists

Long Description
Trying to create an AVM with an ID that already exists

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:66 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:10 [64]13Exception: ERROR (5110): AVM already exists
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateNewAvm(AvmEleme ntHandler.java:170)
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.addEl
ement(AvmElementHa ndler.java:89)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)a
t com.sheer.framework.imo.DataContainer.add(DataContainer.java:154)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoNo
tificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(I
moNotificationUti l.java:293)
```

```
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotifica
tion(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.hand
leNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateAvm.run(Create
Avm.java:113) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

5110

Creating an AVM with a Key That Already Exists

Short Description

Creating an AVM with a key that already exists

Long Description

Trying to create an AVM with a key that already exists

Error Example

```
Command Failed------ com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:68 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:11 [64]13Exception: ERROR (5125): AVM key already exist.
```

```
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valida
teNewAvm(AvmEleme ntHandler.java:193)
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.addEl
ement(AvmElementHa ndler.java:89)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElmentH
andler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.b
eforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at
com.sheer.framework.imo.DataContainer.add(DataContainer.java:154) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(
ImoNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateAvm.run(CreateA
vm.java:113) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
```

```
and.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code
5125

Deleting an AVM That Has VNEs

Short Description
Deleting an AVM that has VNEs

Long Description
Trying to delete an AVM that has VNEs

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:6a [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:12 [64]13Exception: ERROR (5121): Can not delete AVM, AVM
has Devices under it.
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateDeleteAvm(AvmEl ementHandler.java:265)
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.remov
eElement(AvmElemen tHandler.java:281)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:87)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.remove(DataContainer.java:244)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeFromIMO
(ImoNotificationUt il.java:313)
at
```

```
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeIMOsfro
mIMO(ImoNotificati onUtil.java:307)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:162)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.hand
leNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteAvm.deleteAvm(
DeleteAvm.java:155
)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteAvm.run(DeleteA
vm.java:148) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)at
com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code

5121

Restarting an AVM for a Unit That Does Not Exist

Short Description

Restarting an AVM for a unit that does not exist

Long Description

Trying to restart an AVM for a unit that does not exist

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:6c [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:13 [64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.oldcommands.RestartAvm.run(Resta
rtAvm.java:130)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
----------------------------------------------------------------------
```

Error Code

7001

Restarting AVM 99

Short Description

Restarting AVM 99

Long Description

Trying to restart AVM 99

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:6d [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:14 [64]13Exception: ERROR (5114): Restart of AVM 99 is not
possible via DNA. must be done manually.
at
com.sheer.metromission.plugin.bosmanage.oldcommands.RestartAvm.run(Resta
rtAvm.java:134)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

```
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
5114

Restarting a Reserved AVM

Short Description
Restarting a reserved AVM

Long Description
Trying to restart a reserved AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:6e [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:15 [64]13Exception: ERROR (5113): This is a reserved AVM
number. Action not alowed on reserved AVMs.
at
com.sheer.metromission.plugin.bosmanage.oldcommands.RestartAvm.run(Resta
rtAvm.java:138)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
5113

Restarting a Nonexistent AVM

Short Description
Restarting a nonexistent AVM

Long Description
Trying to restart a nonexistent AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
```

```
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:6f [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:16 [64]13Exception: ERROR (5109): AVM does not exist
at
com.sheer.metromission.plugin.bosmanage.oldcommands.RestartAvm.run(Resta
rtAvm.java:143)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
---------------------------------------------------------------------
```

Error Code

5109

Updating an AVM for a Nonexistent Unit

Short Description

Updating an AVM for a nonexistent unit

Long Description

Trying to update an AVM for a nonexistent unit

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:70 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:17 [64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE lementHandler.java:929)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
```

```
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
------------------------------------------------------------------
```

Error Code
7001

Updating a Nonexistent AVM

Short Description
Updating a nonexistent AVM

Long Description
Trying to update a nonexistent AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:71 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:18 [64]13Exception: ERROR (5109): AVM does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE lementHandler.java:932)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
------------------------------------------------------------------
```

Error Code
5109

Moving a VNE from a Nonexistent Unit

Short Description
Moving a VNE from a nonexistent unit

Long Description
Trying to move a VNE from a nonexistent unit

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:72 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:19 [64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE lementHandler.java:929)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
---------------------------------------------------------------------
```

Error Code
7001

Moving a VNE from a Nonexistent AVM

Short Description
Moving a VNE from a nonexistent AVM

Long Description

Trying to move a VNE from a nonexistent AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:73 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:1a [64]13Exception: ERROR (5109): AVM does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE
lementHandler.java:932) at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code

5109

Moving a VNE to the Same AVM

Short Description

Moving a VNE to the same AVM

Long Description

Trying to move a VNE to the same AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:74 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:1b [64]13Exception: ERROR (5130): This device already
resides in this Avm.
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.validateMoveElement(ElementManagementElementHandler.java:931)
```

```
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.moveEleme nt(ElementManagementElementHandler.java:954)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.imObjectMove
(BosManagePluginIm pl.java:306)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
imObjectMove(ImoNo tificationUtil.java:377)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.imObjec
tMove(ImoChangesTo Notifications.java:69)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:178)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code

5130

Creating a VNE for a Nonexistent Unit

Short Description

Creating a VNE for a nonexistent unit

Long Description

Trying to create a VNE for a nonexistent unit

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:75
```

```
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:1c
[64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE
lementHandler.java:929) at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateDevice.run(Cre
ateDevice.java:96) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
---------------------------------------------------------------------
```

Error Code
7001

Creating a VNE in a Nonexistent AVM

Short Description
Creating a VNE in a nonexistent AVM

Long Description
Trying to create a VNE in a nonexistent AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:77 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:1d [64]13Exception: ERROR (5109): AVM does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE
lementHandler.java:932) at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateDevice.run(Cre
ateDevice.java:96)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
---------------------------------------------------------------------
```

Error Code

5109

Creating a VNE in a Reserved AVM

Short Description

Creating a VNE in a reserved AVM

Long Description

Trying to create a VNE in a reserved AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:79 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:1e [64]13Exception: ERROR (5113): This is a reserved AVM
number. Action not alowed on reserved AVMs.
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.validateNewElement(ElementManagementElementHandler.java:371)
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.addElemen t(ElementManagementElementHandler.java:108)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElmentH
andler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.b
eforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at
com.sheer.framework.imo.DataContainer.add(DataContainer.java:154) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(
ImoNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
```

```
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateDevice.run(Cre
ateDevice.java:96) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code

5113

Creating a VNE That Already Exists with the Same Name

Short Description

Creating a VNE that already exists with the same name

Long Description

Trying to create a VNE that already exists with the same name

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:7d
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:20
[64]13Exception: ERROR (5111): An VNE by that name already exists
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.validateNewElement(ElementManagementElementHandler.java:374) at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.addElemen t(ElementManagementElementHandler.java:108)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElmentH
andler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
```

```
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.add(DataContainer.java:154)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(
ImoNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateDevice.run(Cre
ateDevice.java:96) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code

5111

Creating a VNE That Already Exists with the Same IP Address

Short Description
Creating a VNE that already exists with the same IP address

Long Description
Trying to create a VNE that already exists with the same IP address

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:7f [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:21 [64]13Exception: ERROR (5104): An VNE by that IP
already exists
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.validateNewElement(ElementManagementElementHandler.java:377)
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.addElemen t(ElementManagementElementHandler.java:108)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.add(DataContainer.java:154)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoNo
tificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(I
moNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.hand
leNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
```

```
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessCo
mmandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificatio
nUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateDevice.run(Cre
ateDevice.java:96)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)at
com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code
5104

Creating a VNE That Has an Invalid Device Name

Short Description
Creating a VNE that has an invalid device name

Long Description
Trying to create a VNE that has an invalid device name

Error Example

```
Invalid Command Syntax:
java.lang.IllegalArgumentException: Invalid command syntax. Argument
"imobject" is invalid.
```

Error Code
5133

Creating a VNE That Has An Invalid Device Type

Short Description
Creating a VNE that has an invalid device type

Long Description
Trying to create a VNE that has an invalid device type

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:81 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:22 [64]13Exception: ERROR (5105): Unrecognized vendor/type
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.validateNewElement(ElementManagementElementHandler.java:388)
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.addElemen t(ElementManagementElementHandler.java:108)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElmentH
andler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.b
eforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at
com.sheer.framework.imo.DataContainer.add(DataContainer.java:154) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(
ImoNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
```

```
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateDevice.run(Cre
ateDevice.java:96) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

5105

Updating a VNE in a Nonexistent Unit

Short Description

Updating a VNE in a nonexistent unit

Long Description

Trying to update a VNE in a nonexistent unit

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:83 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:23 [64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.validateElementOid(ElementManagementElementHandler.java:902)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

```
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
----------------------------------------------------------------------
```

Error Code
7001

Updating a VNE in a Nonexistent AVM

Short Description
Updating a VNE in a nonexistent AVM

Long Description
Trying to update a VNE in a nonexistent AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:84 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:24 [64]13Exception: ERROR (5109): AVM does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.validateElementOid(ElementManagementElementHandler.java:905)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)at
com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
----------------------------------------------------------------------
```

Error Code
5109

Updating a Nonexistent VNE

Short Description
Updating a nonexistent VNE

Long Description
Trying to update a nonexistent VNE

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:85 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:25 [64]13Exception: ERROR (5103): Agent doesn't exist
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.validateElementOid(ElementManagementElementHandler.java:908) at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
---------------------------------------------------------------------
```

Error Code
5103

Deleting a VNE in a Nonexistent Unit

Short Description
Deleting a VNE in a nonexistent unit

Long Description
Trying to delete a VNE in a nonexistent unit

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:86 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:26 [64]13Exception: ERROR (7001): Unit does not exist
```

```
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE lementHandler.java:929)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteDevice.deleteD
evice(DeleteDevice
.java:144) at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteDevice.run(Del
eteDevice.java:134
)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
-------------------------------------------------------------------
```

Error Code
7001

Deleting a VNE in a Nonexistent AVM

Short Description
Deleting a VNE in a nonexistent AVM

Long Description

Trying to delete a VNE in a nonexistent AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:88 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:27 [64]13Exception: ERROR (5109): AVM does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE lementHandler.java:932)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteDevice.deleteD
evice(DeleteDevice
.java:144) at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteDevice.run(Del
eteDevice.java:134
)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
---------------------------------------------------------------------
```

Error Code

5109

Deleting a Nonexistent VNE

Short Description
Deleting a nonexistent VNE

Long Description
Trying to delete a nonexistent VNE

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:8e [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:28 [64]13Exception: ERROR (5103): Agent doesn't exist
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.validateElementOid(ElementManagementElementHandler.java:908)
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.validateRemoveElement(ElementManagementElementHandler.java:201)
at
com.sheer.metromission.plugin.bosmanage.handlers.ElementManagementElemen
tHandler.removeElement(ElementManagementElementHandler.java:140)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:87)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.remove(DataContainer.java:244)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeFromIMO
(ImoNotificationUt il.java:313)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeIMOsfro
mIMO(ImoNotificati onUtil.java:307)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:162)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
```

```
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificatio
nUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteDevice.deleteD
evice(DeleteDevice
.java:144)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteDevice.run(Del
eteDevice.java:134
)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)at
com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code

5103

Creating an Alias for a Nonexistent VNE

Short Description

Creating an alias for a nonexistent VNE

Long Description

Trying to create an alias for a nonexistent VNE

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:8e [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:2a [64]13Exception: ERROR (5103): Agent doesn't exist
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateAlias.run(Crea
teAlias.java:122)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
```

```
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
----------------------------------------------------------------------
```

Error Code
5103

Creating an Alias for an Element with the Same Name

Short Description
Creating an alias for an element with the same name

Long Description
Trying to create an alias for an element with the same name

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:8f
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:2b
[64]13Exception: ERROR (5111): An VNE by that name already exists
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateAlias.run(Crea
teAlias.java:125)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
5111

Creating a Scope That Already Exists

Short Description
Creating a scope that already exists

Long Description
Trying to create a scope that already exists

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:90
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:2c
[64]13Exception:
java.lang.Exception: This key already exists at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateScope.run(Crea
teScope.java:130) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
) at
com.sheer.metromission.session.Session.processMessage(Session.java:363)a
t com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
None

Creating a Scope with No OID

Short Description
Creating a scope with no OID

Long Description
Trying to create a scope with no OID

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:91
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:2d
[64]13Exception:
java.lang.Exception: Cannot create new Scope. No oid was supplied at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateScope.run(Crea
teScope.java:118) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
```

```
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
None

Creating a User That Already Exists

Short Description
Creating a user that already exists

Long Description
Trying to create a user that already exists

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:92
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:2e
[64]13Exception: ERROR (8001): User already exists
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateUser.run(Creat
eUser.java:141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
-----------------------------------------------------------------
```

Error Code
8001

Creating a User with an Illegal Username or Password

Short Description
Creating a user with an illegal username or password

Long Description
Trying to create a user with an illegal username or password

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:93
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:2f
[64]13Exception: ERROR (8002): Illegal user name or
password - The password can only consist of the following characters:
!"#$%"()*+,-
./0123456789:;=?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvw
xyz{|}
~&<>
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateUser.run(Creat
eUser.java:150)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code

8002

Deleting an Alias for a Nonexistent VNE

Short Description

Deleting an alias for a nonexistent VNE

Long Description

Trying to delete an alias for a nonexistent VNE

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:94 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:30 [64]13Exception: ERROR (5103): Agent doesn’t exist
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteAlias.run(Dele
teAlias.java:122) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
```

```
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
-------------------------------------------------------------------
```

Error Code

5103

Deleting a Reserved Scope

Short Description

Deleting a reserved scope

Long Description

Trying to delete a reserved scope

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:95
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:31
[64]13Exception:
com.sheer.metromission.plugin.security.SecurityViolationException:
Delete reserved scopes is forbidden!!
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteScope.run(Del
eteScope.java:139)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessCom
mand.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:5
5)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at
com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code

None

Deleting a Nonexistent Username

Short Description

Deleting a nonexistent username

Long Description
Trying to delete a nonexistent username

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:96 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:32 [64]13Exception: java.lang.IllegalArgumentException:
User not found in the db
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteUser.run(Delet
eUser.java:134)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
None

Deleting a Reserved Username

Short Description
Deleting a reserved username

Long Description
Trying to delete a reserved username

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:97
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:33
[64]13Exception:
com.sheer.metromission.plugin.security.SecurityViolationException: User
is not a normal user and cannot be deleted
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteUser.run(Delet
eUser.java:139)
```

```
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
None

Updating a Reserved Scope

Short Description
Updating a reserved scope

Long Description
Trying to update a reserved scope

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:98 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:34 [64]13Exception:
com.sheer.metromission.plugin.security.SecurityViolationException:
Update reserved scopes is forbidden!!
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdateScope.run(Upda
teScope.java:122) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
None

Adding Permission for a User with an Administrator Role

Short Description
Adding permission for a user with an administrator role

Long Description
Trying to add a permission for a user with an administrator role

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:99
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:35
[64]13Exception:
com.sheer.metromission.plugin.security.SecurityViolationException: Adding
permission with administrator role is forbidden!!
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdateUser.createPerm
issionEntity(Upda teUser.java:222)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdateUser.run(Updat
eUser.java:178)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
None

Updating a Protected Username

Short Description
Updating a protected username

Long Description
Trying to update a protected username

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:9a [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:36 [64]13Exception:
```

```
com.sheer.metromission.plugin.security.SecurityViolationException:
User:0 is an protected user and cannot be updated except password
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdateUser.run(Updat
eUser.java:145)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code

None

Updating a Username with an Unknown Role

Short Description

Updating a username with an unknown role.

Long Description

Trying to update a username with an unknown role.

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:9b [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:37 [64]13Exception:
com.sheer.metromission.plugin.security.SecurityViolationException: Tried
to add a permission with an unknown role
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdateUser.createPerm
issionEntity(Upda teUser.java:222)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdateUser.run(Updat
eUser.java:178)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

```
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
None

Updating a Nonexistent Username

Short Description
Updating a nonexistent username

Long Description
Trying to update a nonexistent username

Error Example

```
Command Failed------ com.sheer.framework.commands.messages.ExceptionMessage
------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:9c
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:38
[64]13Exception: java.lang.IllegalArgumentException: User
{[BOSUser(Id=666)]} not found in the DB at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdateUser.getUser(Upd
ateUser.java:281
)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdateUser.run(UpdateU
ser.java:121)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComman
d.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55)
at com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
None

Loading an AVM That Is Already Loaded

Short Description
Loading an AVM that is already loaded

Long Description
Trying to load an AVM that is already loaded

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:9f [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:3a [64]13Exception: ERROR (5119): AVM Already Loaded.
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.updat
eElementRegistry(A
vmElementHandler.java:360) at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:98)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.set(DataContainer.java:142)
at com.sheer.framework.imo.IMObject.set(IMObject.java:199) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.updateIMO(Imo
NotificationUtil.j ava:283)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.updateIMOProp
erty(ImoNotificati onUtil.java:276)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotifica
tion(ImoNotificat ionUtil.java:153)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.hand
leNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificatio
nUtil.generateNotification(ManagementNotificationUtil.java:129)
```

```
at
com.sheer.metromission.plugin.bosmanage.oldcommands.LoadAvm.run(LoadAvm.
java:112)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

5119

Loading an AVM in a Nonexistent Unit

Short Description

Loading an AVM in a nonexistent unit

Long Description

Trying to load an AVM in a nonexistent unit

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:a1
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:3b
[64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE
lementHandler.java:929) at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
```

```
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:129)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.LoadAvm.run(LoadAvm.
java:112)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
---------------------------------------------------------------------
```

Error Code
7001

Loading a Nonexistent AVM

Short Description
Loading a nonexistent AVM

Long Description
Trying to load a nonexistent AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:a3
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:3c
[64]13Exception: ERROR (5109): AVM does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE
lementHandler.java:932) at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
```

```
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:129)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.LoadAvm.run(LoadAvm.
java:112)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
-------------------------------------------------------------------
```

Error Code

5109

Unloading an AVM in a Nonexistent Unit

Short Description

Unloading an AVM in a nonexistent unit

Long Description

Trying to unload an AVM in a nonexistent unit

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:a5 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:3d [64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE lementHandler.java:929)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:20)
```

```
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:129)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UnloadAvm.run(Unload
Avm.java:112)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
------------------------------------------------------------------
```

Error Code
7001

Unloading a Nonexistent AVM

Short Description
Unloading a nonexistent AVM

Long Description
Trying to unload a nonexistent AVM

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:a7
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:3e
[64]13Exception: ERROR (5109): AVM does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.valid
ateElementOid(AvmE
```

```
lementHandler.java:932) at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.va
lidateNotification (BOSManageCommandUtil.java:49)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:20)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:129)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UnloadAvm.run(Unload
Avm.java:112) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
---------------------------------------------------------------------
```

Error Code

5109

Updating an Unknown Property in a Protection Group

Short Description

Updating an unknown property in a protection group

Long Description

Trying to update an unknown property in a protection group

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:a9
```

```
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:3f
[64]13Exception: ERROR (4000): Command not supported
at
com.sheer.metromission.plugin.bosmanage.handlers.ProtectionGroupElementH
andler.updateElementRegistry(ProtectionGroupElementHandler.java:75)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:98)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.set(DataContainer.java:142)
at com.sheer.framework.imo.IMObject.set(IMObject.java:199) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.updateIMO(Imo
NotificationUtil.j ava:283)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.updateIMOPrope
rty(ImoNotificati onUtil.java:276)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:153)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code

4000

Updating a Permission for a Nonexistent Username

Short Description
Updating a permission for a nonexistent username

Long Description
Trying to update a permission for a nonexistent username

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:aa
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:40
[64]13Exception:
java.lang.IllegalArgumentException: User {[BOSUser(Id=666)]} not found
in the DB
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdatePermission.get
User(UpdatePermiss ion.java:196)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdatePermission.run(
UpdatePermission. java:146)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code
None

Updating a Permission for a Protected Username

Short Description
Updating a permission for a protected username

Long Description
Trying to update a permission for a protected username

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:ab
```

```
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:41
[64]13Exception:
com.sheer.metromission.plugin.security.SecurityViolationException: Only
regular users" permissions may be updated
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdatePermission.run
(UpdatePermission. java:150)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code

None

Updating a Nonexistent Permission

Short Description

Updating a nonexistent permission

Long Description

Trying to update a nonexistent permission

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:ac
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:42
[64]13Exception: java.lang.IllegalArgumentException: User
{[BOSUser(Id=555)]} not found in the DB at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdatePermission.get
User(UpdatePermiss
ion.java:196) at
com.sheer.metromission.plugin.bosmanage.oldcommands.UpdatePermission.run
(UpdatePermission. java:146)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
```

```
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
```

Error Code

None

Creating a Unit That Already Exists

Short Description

Creating a unit that already exists

Long Description

Trying to create a unit that already exists

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:ad [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:43 [64]13Exception: ERROR (7002): Unit already exists
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.vali
dateNewUnit(UnitEl ementHandler.java:221)
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.addEl
ement(UnitElement Handler.java:102)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElmentH
andler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.add(DataContainer.java:154)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(I
moNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.hand
leNotification(Ge tAndRegisterPluginComponent.java:153)
```

```
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateMC.run(CreateM
C.java:133)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

7002

Creating a Redundant Unit with the Same IP Address

Short Description

Creating a redundant unit with the same IP address

Long Description

Trying to create a redundant unit with the same IP address

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:b1 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:45 [64]13Exception: ERROR (7002): Unit already exists
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.vali
dateNewUnit(UnitEl ementHandler.java:221)
```

```
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.addE
lement(UnitElement Handler.java:102)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElmentH
andler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.b
eforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at
com.sheer.framework.imo.DataContainer.add(DataContainer.java:154) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(
ImoNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateMC.run(CreateM
C.java:133) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComma
nd.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
```

```
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

7005

Creating a Unit with an Invalid IP Address

Short Description

Creating a unit with an invalid IP address

Long Description

Trying to create a unit with an invalid IP address

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:b5
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:47
[64]13Exception: ERROR (5135): Illegal IP address.
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.valid
ateNewUnit(UnitEl ementHandler.java:227)
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.addE
lement(UnitElement
Handler.java:102) at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.add(DataContainer.java:154)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(
ImoNotificationUti l.java:293)
```

```
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComma
nd.java:68)at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateMC.run(CreateM
C.java:133) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

None

Creating a Unit with a Nonexistent Protection Group

Short Description

Creating a unit with a nonexistent protection group

Long Description

Trying to create a unit with a nonexistent protection group

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
```

```
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:b7 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:48 [64]13Exception: ERROR (1100000001): No protection
group with that name exists.
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.vali
dateNewUnit(UnitEl ementHandler.java:236)
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.addE
lement(UnitElement Handler.java:102)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.add(DataContainer.java:154)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoNo
tificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(I
moNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.hand
leNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessCo
mmandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificatio
nUtil.generateNotification(ManagementNotificationUtil.java:110)
```

```
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateMC.run(CreateM
C.java:133)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)at
com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code

1100000001

Creating a Polling Group That Already Exists

Short Description

Creating a polling group that already exists

Long Description

Trying to create a polling group that already exists

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:b9 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:49 [64]13Exception: ERROR (9003): Polling group already
exists
at
com.sheer.metromission.plugin.bosmanage.handlers.PollingGroupElementHand
ler.validateNewPollingGroup(PollingGroupElementHandler.java:268)
at
com.sheer.metromission.plugin.bosmanage.handlers.PollingGroupElementHand
ler.addElement(PollingGroupElementHandler.java:100)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElmentH
andler.updateElement(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.b
eforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at
```

```
com.sheer.framework.imo.DataContainer.add(DataContainer.java:154) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(
ImoNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotifica
tion(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreatePollingGroup.r
un(CreatePollingGroup.java:109)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
) at
com.sheer.metromission.session.Session.processMessage(Session.java:363)a
t com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code
9003

Creating a Redundant Unit That Already Exists with the Same IP Address

Short Description
Creating a redundant unit that already exists with the same IP address

Long Description
Trying to create a redundant unit that already exists with the same IP address

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:bd
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:4b
[64]13Exception: ERROR (7002): Unit already exists
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.valid
ateNewUnit(UnitEl ementHandler.java:221)
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.addE
lement(UnitElement
Handler.java:102)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)a
t com.sheer.framework.imo.DataContainer.add(DataContainer.java:154)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoNo
tificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(I
moNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
```

```
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateRedundantMC.ru
n(CreateRedundantM C.java:140)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

7005

Creating a Redundant Unit That Already Exists as a Unit

Short Description

Creating a redundant unit that already exists as a unit

Long Description

Trying to create a redundant unit that already exists as a unit

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:c3
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:4e
[64]13Exception: ERROR (7002): Unit already exists
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.valid
ateNewUnit(UnitEl ementHandler.java:221)
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.addE
lement(UnitElement
Handler.java:102) at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
```

```
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.add(DataContainer.java:154)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(
ImoNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateRedundantMC.ru
n(CreateRedundantM C.java:140)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

7002

Creating a Redundant Unit with a Nonexistent Protection Group

Short Description

Creating a redundant unit with a nonexistent protection group

Long Description

Trying to create a redundant unit with a nonexistent protection group

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:c7 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:50 [64]13Exception: ERROR (1100000001): No protection
group with that name exists.
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.vali
dateNewUnit(UnitEl ementHandler.java:236)
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.addE
lement(UnitElement Handler.java:102)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at
com.sheer.framework.imo.DataContainer.add(DataContainer.java:154) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(I
moNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotifica
tion(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.hand
leNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
```

```
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessCo
mmandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificatio
nUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateRedundantMC.run
(CreateRedundantM C.java:140)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)at
com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code
1100000001

Deleting a Nonexistent Unit

Short Description
Deleting a nonexistent unit

Long Description
Trying to delete a nonexistent unit

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:c9
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:51
[64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteMC.deleteMc(De
leteMC.java:148)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteMC.run(DeleteM
C.java:136) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComma
nd.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
```

```
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
----------------------------------------------------------------------
```

Error Code

7001

Deleting the Gateway

Short Description

Deleting the gateway

Long Description

Trying to delete the gateway

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:ca [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:52 [64]13Exception: ERROR (7004): Cannot delete Unit, Unit
is a Gateway.
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.vali
dateRemoveUnit(Uni tElementHandler.java:311)
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.remo
veElement(UnitElem entHandler.java:246)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:87)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.remove(DataContainer.java:244)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeFromIMO
(ImoNotificationUt il.java:313)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeIMOsfro
mIMO(ImoNotificati onUtil.java:307)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:162)
```

```
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.hand
leNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComma
nd.java:68)at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteMC.deleteMc(De
leteMC.java:150)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteMC.run(DeleteM
C.java:136) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComma
nd.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)at
com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code

7004

Deleting a Unit That Has AVMs

Short Description

Deleting a unit that has AVMs

Long Description

Trying to delete a unit that has AVMs

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:d0 [64]13Destination: (CL.TS-64.103.124.248 [5]-
```

```
0:0:0:0:0:0:0:55 [64]13Exception: ERROR (7003): Cannot delete Unit, Unit
has AVMs under it.
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.vali
dateRemoveUnit(Uni tElementHandler.java:314)
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.remo
veElement(UnitElem entHandler.java:246)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:87)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.b
eforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at
com.sheer.framework.imo.DataContainer.remove(DataContainer.java:244) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeFromIMO
(ImoNotificationUt il.java:313)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeIMOsfrom
IMO(ImoNotificati onUtil.java:307)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:162)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessCo
mmandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificatio
nUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteMC.deleteMc(De
leteMC.java:150)
```

```
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteMC.run(DeleteM
C.java:136) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
) at
com.sheer.metromission.session.Session.processMessage(Session.java:363)a
t com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code
7003

Deleting a VNE

Short Description
Deleting a VNE

Long Description
Trying to delete a VNE

Error Example

```
<?xml version="1.0" encoding="UTF-8"?><command name="Delete"><param
name="oids"><value>{[MCNetwork][MCVM(IP=10.104.253.102)][Avm(AvmNumber=3
05)][ElementManagement(Key=radtest-75002-4)]}</value></param></command>

Received buffer --
<?xml version="1.0" encoding="UTF-8"?>
<ISystemError type=”ISystemError” instance_id=”1>
<ID type="Oid">{[SystemError(Code=1000)]}</ID>
<Description type="String">ERROR (1000): General error, Exception:
java.lang.NullPointerException</Description>
<ErrorStackTrace type="java.lang.String_Array">

<java.lang.String>com.sheer.metromission.plugin.bosmanage.oldcommands.De
leteDevice.deleteDevice(DeleteDevice.java:224)</java.lang.String>

<java.lang.String>com.sheer.metromission.plugin.bosmanage.oldcommands.De
leteDevice.run(Del eteDevice.java:212)</java.lang.String>
java.lang.String>com.sheer.framework.commands.StatelessCommand.localExecu
te(StatelessComm and.java:68)</java.lang.String>

<java.lang.String>com.sheer.framework.commands.Command.execute(Command.j
ava:63)</java.lang
.String>

<java.lang.String>com.sheer.metromission.session.CommandEntry.execute(Co
mmandEntry.java:55
)</java.lang.String>

<java.lang.String>com.sheer.metromission.session.Session.processMessage(
Session.java:368)<
/java.lang.String>
```

```
<java.lang.String>com.sheer.system.agentshell.AgentBase.run(AgentBase.ja
va:232)</java.lang
.String>

<java.lang.String>com.sheer.system.os.services.scheduler.OSAgent.run(OSAg
ent.java:107)</java.lang.String>

<java.lang.String>com.sheer.system.os.util.ThreadPool$OSThread.run(Thread
Pool.java:272)</java.lang.String>
</ErrorStackTrace>
</ISystemError>
```

Error Code

1000

Deleting a Polling Group That Is Being Used by a Device

Short Description

Deleting a polling group that is being used by a device

Long Description

Trying to delete a polling group that is being used by a device

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:de [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:5c [64]13Exception: ERROR (9001): Polling group is in use
by one or more devices. -
{[MCNetwork][MCVM(IP=1.1.1.1)][Avm(AvmNumber=123)][ElementManagement(Key=
yael)]} at
com.sheer.metromission.plugin.bosmanage.handlers.PollingGroupElementHand
ler.removeElement( PollingGroupElementHandler.java:166)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElmentH
andler.updateElem ent(AbstractBosManageElmentHandler.java:87)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)a
t com.sheer.framework.imo.DataContainer.remove(DataContainer.java:244)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeFromIMO(
ImoNotificationUt il.java:313)
```

```
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeIMOsfrom
IMO(ImoNotificati onUtil.java:307)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotifica
tion(ImoNotificat ionUtil.java:162)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.hand
leNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeletePollingGroup.d
eleteGroup(DeleteP ollingGroup.java:142)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeletePollingGroup.r
un(DeletePollingGr oup.java:133)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
) at
com.sheer.metromission.session.Session.processMessage(Session.java:363)a
t com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code
9001

Deleting the Default Polling Group

Short Description
Deleting the default polling group

Long Description
Trying to delete the default polling group

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:e8 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:61 [64]13Exception: ERROR (9002): Deletion of "default"
polling group is not allowed.
at
com.sheer.metromission.plugin.bosmanage.handlers.PollingGroupElementHandl
er.removeElement( PollingGroupElementHandler.java:126)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:87)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)a
t com.sheer.framework.imo.DataContainer.remove(DataContainer.java:244)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeFromIMO
(ImoNotificationUt il.java:313)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeIMOsfro
mIMO(ImoNotificati onUtil.java:307)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:162)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
```

```
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeletePollingGroup.d
eleteGroup(DeleteP ollingGroup.java:142)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeletePollingGroup.r
un(DeletePollingGr oup.java:133)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)at
com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code

9002

Deleting a Nonexistent Redundant Unit

Short Description

Deleting a nonexistent redundant unit

Long Description

Trying to delete a nonexistent redundant unit

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:ea [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:62 [64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.vali
dateElementOid(Uni tElementHandler.java:965)
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.vali
dateRemoveUnit(Uni tElementHandler.java:308)
at
com.sheer.metromission.plugin.bosmanage.handlers.UnitElementHandler.remo
veElement(UnitElem entHandler.java:246)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:87)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
```

```
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.remove(DataContainer.java:244)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeFromIMO
(ImoNotificationUt il.java:313)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeIMOsfro
mIMO(ImoNotificati onUtil.java:307)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:162)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.hand
leNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificatio
nUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteRedundantMC.de
leteRedundantMC(De leteRedundantMC.java:153)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteRedundantMC.run
(DeleteRedundantM C.java:145)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

7001

Failover for a Nonexistent Unit

Short Description
Failover for a nonexistent unit

Long Description
Failover for a nonexistent unit

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:ec [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:63 [64]13Exception: ERROR (7001): Unit does not exist
at
com.sheer.metromission.plugin.bosmanage.oldcommands.McManualFailover.fai
loverMc(McManualFa ilover.java:188)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.McManualFailover.run
(McManualFailover. java:176)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
---------------------------------------------------------------------
```

Error Code
7001

Restarting a Nonexistent Unit

Short Description
Restarting a nonexistent unit

Long Description
Trying to restart a nonexistent unit

Error Example
```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:ed [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:64 [64]13Exception: ERROR (7001): Unit does not exist
```

```
at
com.sheer.metromission.plugin.bosmanage.oldcommands.MCRestart.restartMc(
MCRestart.java:174
)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.MCRestart.run(MCResta
rt.java:161) at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
----------------------------------------------------------------------
```

Error Code

7001

Invalid Transport Uplink Command

Short Description

Invalid transport **uplink** command

Long Description

Invalid transport **uplink** command

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:ee
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:65
[64]13Exception: ERROR (2000): Execution failed
at
com.sheer.metromission.plugin.bosmanage.oldcommands.TransportUplinkCtl.r
un(TransportUplink
Ctl.java:276)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
----------------------------------------------------------------------
```

Error Code
None

Creating a Static Topological Link with a Nonexistent VNE

Short Description
Creating a static topological link with a nonexistent VNE

Long Description
Trying to create a static topological link with a nonexistent VNE

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:ef [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:66 [64]13Exception: ERROR (5103): Agent doesn't exist
at
com.sheer.metromission.plugin.bosmanage.handlers.StaticTopologyLinkManag
ementHandler.validateNewElement(StaticTopologyLinkManagementHandler.java
:196)
at
com.sheer.metromission.plugin.bosmanage.handlers.StaticTopologyLinkManag
ementHandler.addElement(StaticTopologyLinkManagementHandler.java:127)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at
com.sheer.framework.imo.DataContainer.add(DataContainer.java:154) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoN
otificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(
ImoNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotifica
tion(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.hand
leNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:40)
```

```
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateStaticTopologyMan
agement.run(Update
StaticTopologyManagement.java:136)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificatio
nUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateStaticTopology
Link.run(CreateSta ticTopologyLink.java:94)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)at
com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code

5103

Creating a Static Topological Link That Already Exists

Short Description

Creating a static topological link that already exists

Long Description

Trying to create a static topological link that already exists

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:f3 [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:68 [64]13Exception: ERROR (2104): Device and port are the
same on both sides
at
com.sheer.metromission.plugin.bosmanage.handlers.StaticTopologyLinkManag
ementHandler.addElement(StaticTopologyLinkManagementHandler.java:133)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:71)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage
```

```
PluginImpl.java:251) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.b
eforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC
hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)a
t com.sheer.framework.imo.DataContainer.add(DataContainer.java:154)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addToIMO(ImoNo
tificationUtil.ja va:298)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.addIMOstoIMO(I
moNotificationUti l.java:293)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:158)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.upd
ate(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateStaticTopologyMan
agement.run(Update StaticTopologyManagement.java:136)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessCo
mmandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificatio
nUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.CreateStaticTopology
Link.run(CreateSta ticTopologyLink.java:94)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
) at
com.sheer.metromission.session.Session.processMessage(Session.java:363)a
t com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
```

Error Code

2102

Deleting a Nonexistent Static Topological Link

Short Description

Deleting a nonexistent static topological link

Long Description

Trying to delete a nonexistent static topological link

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:f6
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:6a
[64]13Exception: ERROR (2103): Ports Not Linked to Each Other.
at
com.sheer.metromission.plugin.bosmanage.handlers.StaticTopologyLinkManage
mentHandler.validateElementOid(StaticTopologyLinkManagementHandler.java:
254)
at
com.sheer.metromission.plugin.bosmanage.handlers.StaticTopologyLinkManage
mentHandler.validateRemove(StaticTopologyLinkManagementHandler.java:245)
at
com.sheer.metromission.plugin.bosmanage.handlers.StaticTopologyLinkManage
mentHandler.removeElement(StaticTopologyLinkManagementHandler.java:223)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:87)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.b
eforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at
com.sheer.framework.imo.DataContainer.remove(DataContainer.java:244)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeFromIMO
(I
moNotificationUtil.java:313) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.removeIMOsfro
mIMO(ImoNotificati onUtil.java:307)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:162)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
```

```
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateStaticTopologyMan
agement.run(Update StaticTopologyManagement.java:136)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:110)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteStaticTopology
Link.deleteLink(De leteStaticTopologyLink.java:130)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.DeleteStaticTopology
Link.run(DeleteSta ticTopologyLink.java:121)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
```

Error Code

2103

Creating a Topological Link on a Nonexistent Port

Short Description

Creating a topological link on a nonexistent port

Long Description

Trying to create a topological link on a nonexistent port

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:f9
[64]13Destination: (CL.TS-64.103.124.248 [5]-0:0:0:0:0:0:0:6b
[64]13Exception:
java.lang.Exception: oid
```

```
{[ManagedElement(Key=PE-
West)][PhysicalRoot][Chassis][Slot(SlotNum=1)][Module][Port(PortNumber=Fa
stEthernet1/2)]} does not exist
at com.sheer.metrocentral.framework.maps.command.Get.get(Get.java:219) at
com.sheer.metrocentral.framework.maps.command.Get.get(Get.java:236) at
com.sheer.metrocentral.framework.maps.command.Get$StartStateHandler.hand
le(Get.java:571) at
com.sheer.framework.commands.MultiStateCommand.localExecute(MultiStateCom
mand.java:59) at
com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.system.agentshell.components.cre.CommandRunEnvironment.handleE
xecuteMessage
(CommandRunEnvironment.java:305) at
com.sheer.system.agentshell.components.cre.CommandRunEnvironment$Execute
MessageHandler.handle(CommandRunEnvironment.java:433)
at
com.sheer.system.agentshell.components.cre.CommandRunEnvironment.process
Message
(CommandRunEnvironment.java:183)
at com.sheer.metrocentral.framework.da.DA.processMessage(DA.java:319)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
----------------------------------------------------------------------
```

Error Code

None

Restarting an AVM When It Is Down

Short Description

Restarting an AVM when it is down

Long Description

Trying to restart an AVM when it is down

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:fd [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:6d [64]13Exception: ERROR (5120): AVM Not Loaded.
at
com.sheer.metromission.plugin.bosmanage.oldcommands.RestartAvm.run(Resta
rtAvm.java:148)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
```

```
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
at com.sheer.system.os.util.ThreadPool$OSThread.run(ThreadPool.java:272)
-------------------------------------------------------------------
```

Error Code

5120

Unloading an AVM That Has Already Been Unloaded

Short Description

Unloading an AVM that has already been unloaded

Long Description

Trying to unload an AVM that has already been unloaded

Error Example

```
Command Failed------
com.sheer.framework.commands.messages.ExceptionMessage ------
Exception Message: 13Source: 13(MM.SA-64.103.124.248 [6813261]-
0:0:0:0:0:0:0:fe [64]13Destination: (CL.TS-64.103.124.248 [5]-
0:0:0:0:0:0:0:6e [64]13Exception: ERROR (5120): AVM Not Loaded.
at
com.sheer.metromission.plugin.bosmanage.handlers.AvmElementHandler.updat
eElementRegistry(A vmElementHandler.java:363)
at
com.sheer.metromission.plugin.bosmanage.handlers.AbstractBosManageElment
Handler.updateElem ent(AbstractBosManageElmentHandler.java:98)
at
com.sheer.metromission.plugin.bosmanage.BosManagePluginImpl.beforeProper
tyChange(BosManage PluginImpl.java:251)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil$CallBackImpl.
beforePropertyChange(ImoNotificationUtil.java:360)
at
com.sheer.metromission.util.getcommand.ImoChangesToNotifications.beforeP
ropertyChange(ImoC hangesToNotifications.java:52)
at
com.sheer.framework.imo.IMObject.beforePropertyChange(IMObject.java:326)
at com.sheer.framework.imo.DataContainer.set(DataContainer.java:142)
at com.sheer.framework.imo.IMObject.set(IMObject.java:199) at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.updateIMO(Imo
NotificationUtil.j ava:283)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.updateIMOProp
erty(ImoNotificati onUtil.java:276)
at
com.sheer.metromission.util.getcommand.ImoNotificationUtil.handleNotific
ation(ImoNotificat ionUtil.java:153)
at
com.sheer.metromission.util.getcommand.GetAndRegisterPluginComponent.han
dleNotification(Ge tAndRegisterPluginComponent.java:153)
```

```
at
com.sheer.metromission.plugin.bosmanage.commands.BOSManageCommandUtil.up
date(BOSManageComm andUtil.java:40)
at
com.sheer.metromission.plugin.bosmanage.commands.UpdateBosManage.run(Upd
ateBosManage.java:
141)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58) at
com.sheer.framework.commands.StatelessCommandExecuter.executeStatelessCo
mmand(StatelessCom mandExecuter.java:58)
at
com.sheer.framework.commands.StatelessCommandExecuter.execute(StatelessC
ommandExecuter.jav a:94)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.ManagementNotificati
onUtil.generateNotification(ManagementNotificationUtil.java:129)
at
com.sheer.metromission.plugin.bosmanage.oldcommands.UnloadAvm.run(Unload
Avm.java:112)
at
com.sheer.framework.commands.StatelessCommand.localExecute(StatelessComm
and.java:68)
at com.sheer.framework.commands.Command.execute(Command.java:58)
at
com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:55
)
at
com.sheer.metromission.session.Session.processMessage(Session.java:363)
at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:232)
at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:107)
```

Error Code

5120

5.11.6 BQL Command Output Changes Since Prime Network 3.8

Table 5-19 lists the changes in error messages displayed in the BQL command output for certain workflows since Prime Network 3.8.

Table 5-19 Changes in BQL Command Output since Prime Network

Scenario	Command Output in Cisco ANA 3.7.3	Command Output in Prime Network 3.8
Timeout	Commands sent to device 3750E-24TD-AGG3: Error executing command: <?xml version="1.0" encoding="UTF-8"?> <command name="11"> <param name="oid"> <value>{[ManagedElement(Key=3750E-24TD-AGG3)]}</value> </param> </command> Task: Execute BQL(1533). Reason: result contains no empty string nor IMO, IMO array or OID. Result: Command got timeout ------ com.sheer.framework.commands.messages.TimeoutMessage ------ com.sheer.framework.commands.messages.TimeoutMessage, source=(MM.SA-35 [11])-0:0:0:0:0:0:b:11 [64], destination=(CL.TS-0 [30][64.103.121.213])-0:0:0:0:0:0:0:d [64], id=0 ---------- ----------	Commands sent to device 3750E-24TD-AGG3: Error executing command: <?xml version="1.0" encoding="UTF-8"?> <command name="11"> <param name="oid"> <value>{[ManagedElement(Key=3750E-24TD-AG G3)]}</value> </param> </command> Task: Execute BQL(3014). Reason: result contains no empty string nor IMO, IMO array or OID. Result: Command got timeout java.lang.RuntimeException: Command got timeout ---------- ----------

Command General Failure	Error executing command: <?xml version="1.0" encoding="UTF-8"?> <command name="ForceClear"> <param name="oids"> <value>{[NewAlarm(Id=450001)]}</value> </param> </command> Task: Execute BQL(1502). Reason: result contains no empty string nor IMO, IMO array or OID. Result: Command Failed ------ com.sheer.framework.commands.messages.ExceptionMessage ------ Exception Message: 13Source: 13(MM.SA-35 [11])-0:0:0:0:0:0:0:9:da [64]13Destination: (CL.TS-0 [30][64.103.121.213])-0:0:0:0:0:0:0:0:3 [64]13Exception: ERROR (2001): Execution failed. Tickets are in use. Please try again. at com.sheer.metromission.plugin.newalarm.commands.AbstractUpdateTicketCommand.run(Abstract UpdateTicketCommand.java:52) at com.sheer.framework.commands.StatelessCommand.localExecute(StatelessCommand.java:68) at com.sheer.framework.commands.Command.execute (Command.java:64) at com.sheer.metromission.session.CommandEntry. execute(CommandEntry.java:71) at com.sheer.metromission.session.Session.processMessage(Session.java:415) at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:280) at com.sheer.system.os.services.scheduler.OSAgent.run(OSAgent.java:157) at java.util.concurrent.ThreadPoolExecutor$Worker.runTask(ThreadPoolExecutor.java:886) at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:908) at java.lang.Thread.run(Thread.java:619) ---------- ---------- ----------	Error executing command: <?xml version="1.0" encoding="UTF-8"?> <command name="ForceClear"> <param name="oids"> <value>{[NewAlarm(Id=450001)]}</value> </param> </command> Task: Execute BQL(3012). Reason: result contains no empty string nor IMO, IMO array or OID. Result: Command Failed java.lang.RuntimeException: ERROR (2001): Execution failed. Tickets are in use. Please try again. com.sheer.metromission.plugin.newalarm.commands.AbstractUpdateTicketCommand.run (AbstractUpdate TicketCommand.java:52) com.sheer.framework.commands.StatelessCommand.localExecute(StatelessCommand.java:68) com.sheer.framework.commands.Command.execute(Co mmand.java:64) com.sheer.metromission.session.CommandEntry.exe cute(CommandEntry.java:71) com.sheer.metromission.session.Session.processMessage(Session.java:415) com.sheer.system.agentshell.AgentBase.run(Agent Base.java:280) com.sheer.system.os.services.scheduler.OSAgent. run(OSAgent.java:157) java.util.concurrent.ThreadPoolExecutor$Worker. runTask(ThreadPoolExecutor.java:886) java.util.concurrent.ThreadPoolExecutor$Worker. run(ThreadPoolExecutor.java:908) java.lang.Thread.run(Thread.java:619) ---------- ----------

Scenario	Command Output in Cisco ANA 3.7.3	Command Output in Prime Network 3.8
Command General Failure (continued)	Unknown RuntimeException caught in task: Execute BQL(1502), RuntimeException message: No empty result, nor IMO, IMO array or OID in result: Command Failed ------ com.sheer.framework.commands.messages.Exception Message ------ Exception Message: 13Source: 13(MM.SA-35 [11])-0:0:0:0:0:0:9:da [64]13Destination: (CL.TS-0 [30][64.103.121.213])-0:0:0:0:0:0:0:3 [64]13Exception: ERROR (2001): Execution failed. Tickets are in use. Please try again. at com.sheer.metromission.plugin.newalarm.commands.AbstractUpdateTicketCommand.run(Abstract UpdateTicketCommand.java:52) at com.sheer.framework.commands.StatelessCommand.localExecute(StatelessCommand.java:68) at com.sheer.framework.commands.Command.execute (Command.java:64) at com.sheer.metromission.session.CommandEntry.execute(CommandEntry.java:71) at com.sheer.metromission.session.Session.processMessage(Session.java:415) at com.sheer.system.agentshell.AgentBase.run(AgentBase.java:280) at com.sheer.system.os.services.scheduler.OSAgent.run (OSAgent.java:157) at java.util.concurrent.ThreadPoolExecutor$Worker.runTask(ThreadPoolExecutor.java:886) at java.util.concurrent.ThreadPoolExecutor$Worker.run (ThreadPoolExecutor.java:908) at java.lang.Thread.run(Thread.java:619) ---------- ---------- ---------- ----------	

Scenario	Command Output in Cisco ANA 3.7.3	Command Output in Prime Network 3.8
Invalid Command Syntax	Commands sent to device c4-agg1: Error executing command: <?xml version="1.0" encoding="UTF-8"?> <command name="aaa2"> <param name="oid"> <value>{[ManagedElement(Key=c4-agg1)]}</value> </param> </command> Task: Execute BQL(1504). Reason: result contains no empty string nor IMO, IMO array or OID. Result: Invalid Command Syntax. java.lang.IllegalArgumentException: Invalid command syntax. Argument 'cid' is invalid. ---------- ---------- Unknown RuntimeException caught in task: Execute BQL(1504), RuntimeException message: No empty result,nor IMO, IMO array or OID in result: Invalid Command Syntax. java.lang.IllegalArgumentException:Invalid command syntax. Argument 'cid' is invalid. ---------- ----------	Commands sent to device c4-agg1: Error executing command: <?xml version="1.0" encoding="UTF-8"?> <command name="aaa2"> <param name="oid"> <value>{[ManagedElement(Key=c4-agg1)]}</value> </param> </command> Task: Execute BQL(3006). Reason: result contains no empty string nor IMO, IMO array or OID. Result: Invalid Command Syntax. java.lang.RuntimeException: Invalid Command Syntax. java.lang.IllegalArgumentException: Invalid command syntax. Argument 'cid' is invalid. ---------- ----------

Invalid Result	Commands sent to device ME-4924-10GE: Error executing command: <?xml version="1.0" encoding="UTF-8"?> <command name="GetElementManagementFromManagedElement Oid"> <param name="oid"> <value>{[ManagedElement(Key=ME-4924-10GE)]} </value> </param> <param name="rs"> <value><key name="MCNetworkNoProperties"> <entry name="depth">100</entry> <entry name="register">false</entry> <entry name="cachedResultAcceptable">false</entry> <key name="requiredProperties"> <key name="com.sheer.imo.management.IPowerDrillDa ta"> <entry name="*"/> </key> <key name="com.sheer.imo.management.IMC"> <entry name="Avms"/> </key> <key name="com.sheer.imo.management.IMCNetwork"> <entry name="MetroCentrals"/> </key> <key name="com.sheer.imo.management.IAvm"> <entry name="ElementManagements"/> <entry name="AvmKey"/> </key> </key> </key></value> </param> </command> Task: Execute BQL(1602). Reason: result contains no empty string nor IMO, IMO array or OID. Result: Hello World ---------- ---------- Unknown RuntimeException caught in task: Execute BQL(1602), RuntimeExceptionmessage: No empty result,nor IMO, IMO array or OID in result: My Invalid Result ---------- ----------	Commands sent to device ME-4924-10GE: Error executing command: <?xml version="1.0" encoding="UTF-8"?> <command name="GetElementManagementFromManagedElementOid "> <param name="oid"> <value>{[ManagedElement(Key=ME-4924-10GE)]}</value> </param> <param name="rs"> <value><key name="MCNetworkNoProperties"> <entry name="depth">100</entry> <entry name= "register">false</entry> <entry name="cachedResultAcceptable ">false</entry> <key name="requiredProperties"> <key name="com.sheer.imo.management.IPowerDrillData"> <entry name="*"/> </key> <key name="com.sheer.imo.management.IMC"> <entry name="Avms"/> </key> <key name="com.sheer.imo.management.IMCNetwork"> <entry name="MetroCentrals"/> </key> <key name="com.sheer.imo.management.IAvm"> <entry name="ElementManagements "/> <entry name="AvmKey"/> </key> </key> </key></value> </param> </command> Task: Execute BQL(2802). Reason: result contains no empty string nor IMO, IMO array or OID. Result: Command Failed java.lang.RuntimeException: No empty result, nor IMO, IMO array or OID: My Invalid Result ----------

Scenario	Command Output in Cisco ANA 3.7.3	Command Output in Prime Network 3.8
Preview for Gateway Command	Task: Execute BQL(1721), sent gateway command: <?xml version="1.0" encoding="UTF-8"?> <command name="ForceClear"> <param name="oids"> <value>{[NewAlarm(Id=450001)]}</value> </param> </command> Received the result: <IMO/> ---------- ----------	Error executing command: <?xml version="1.0" encoding="UTF-8"?> <command name="ForceClear"> <param name="oids"> <value>{[NewAlarm(Id=450001)]}</value> </param> </command> Task: Execute BQL(3008) Reason: Preview mode is supported for Command Builder scripts only ---------- ----------

6 Event Notification Service

This section contains the following topics:

6.1 Using Event Notification Service

Cisco Prime Network supports an Event Notification Service that generates traps or e-mails and sends notifications to the SNMP adapter in the OSS application. Notifications are created for:

- Network events (traps, syslogs, and service events), including events from unmanaged devices
- Non-network events, such as system, security, and provisioning
- Actionable events and non-actionable (standard) events
- Tickets and ticket updates

The events and tickets from devices are converted into either SNMPv1 or SNMPv2 notifications and are formatted according to the CISCO-EPM-NOTIFICATION-MIB.

The key features of Event Notification Service are:

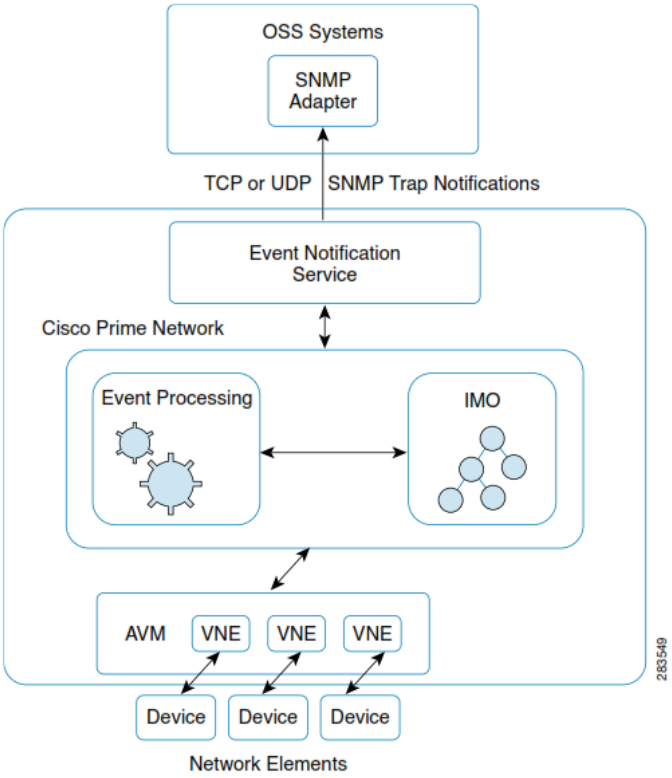
- Allows you to subscribe to notifications, modify the subscription filters, and remove subscriptions.
  The subscriptions are persistent and remain after the system is restarted. You can have multiple subscriptions at the same time. For TCP type of connection, you can have one subscription per destination; where destination refers to IP address and port. For UDP type of connection, you can have more than one subscription per destination.

Note We recommend that the total number of subscriptions not exceed 10.

- Provides ways for remote clients to receive notifications on any port via User Datagram Protocol (UDP) or TCP. It sends notifications to one or multiple user-defined ports using either TCP or UDP.
- Distributes notifications to the respective remote clients according to the configured preferences.
- Subscribes to change notifications for the configured preferences of remote clients. As the preferences are updated, the Event Notification Service is notified and it dynamically changes its behavior at runtime to suit the new preferences.

Figure 6-1 illustrates the relationships among the SNMP adapter on your remote clients, the Event Notification Service on the Prime Network server, and your network devices.

Figure 6-1 Using SNMP to Receive Notifications



Note If you deployed the Cisco ANA2Netcool (AVM80) solution to integrate with CIC or IBM Tivoli (Netcool), contact your Cisco representative for migration support.

Additional Reading

- Review *Cisco Prime Network 4.2.2 Administrator Guide* to understand the Prime Network user roles and scopes and to configure an Event Notification Service.
- Review *Cisco Prime Network 4.2.2 User Guide* to understand the fault management implementation in Prime Network.
- See the *Cisco Prime Network Information Model* Javadoc to understand the IMO for the Event Notification Service. This document is available on the Prime Network Technology Center website. You must have a Cisco.com account with partner level access, or you must be a Prime Network licensee to access this website.

6.1.1 Supported Notification Services in Prime Network

You can subscribe to the following types of notifications:

- Raw Event Notification Service—A notification service supported by Prime Network that maps incoming event notifications (for example, syslogs and traps) into a normalized trap format (EPM-NOTIFICATION-MIB) and generates notifications.
- Event Notification Service—A notification service supported by Prime Network that generates EPM traps and supports sending actionable and non-actionable event notifications, tickets, ticket updates and EMS-generated internal events. For example:
  - Network events: syslogs, traps, service events, tickets, and ticket updates.
  - EMS internal events: security events, system events, and provisioning events.

See Fault Management Terms, page 22 to understand the fault management terms, such as actionable events, nonactionable events, and so on.

6.1.2 Supported Filters

You can apply the following filters in the Event Notification Service:

- Source filter—Receive event notifications generated by managed and unmanaged network elements. You can restrict notifications to a subset of your choice: select all source IP address, select all the network elements that Prime Network manages, select all the network elements of specific types, exclude devices of your choice, select a specific set of devices..
- Ticket or event filter—Define a filter for either events or ticket notifications.
- Event category—Define a filter based on the category of event. The event categories can be one of the following: service, syslog, SNMPv1 trap, SNMPv2 trap, SNMPv3 trap, provisioning, system, security, ticket, ticket update.
  In the ticketupdate filter, a notification is sent for every change to the ticket, and the user can specify the properties for which updates must be received; for example, it can be for LASTMODIFICATIONTIME, AGGREGATEDSEVERITYENUM, AFFECTEDDEVICESCOUNT, AGGREGATEDACKSTATEENUM,EVENTCOUNT, ALARMCOUNT, REDUCTIONCOUNT,DUPLICATIONCOUNT, NOTE, DESCRIPTION, USEROPERATIONS, SEVERITYENUM, LASTEVENTTIME, ACKSTATEENUM, or LATESTSTATE.
- Event type—Define a filter based on the name as defined in Prime Network. You can use the include or exclude parameter when defining the filter. You can use this parameter for event and ticket notifications.

For the event type filter, you must enter the integer value that is associated with the event type. This value is available in the alarm-types.xml file under $*NETWORKHOME*/Main/registry. By default, $*NETWORKHOME* is /export/home/network310. The user-defined events are available in the site.xml file.

If you want details on specific event types, you can also refer to files mm_events.xml for security and system events and send-alarm-msg-util.xml for syslog, traps, and service alarms.

For example, if you want to filter the BGP Link Down event type, you must enter the value 1221.

To include all event type, you can use the wildcard -1.

- Severity—Define a filter based on the severity as perceived in Prime Network. You can use this for event and tickets notifications. The severities can be one of the following: Indeterminate, Information, Cleared, Warning, Minor, Major, or Critical.

Table 6-1 defines the supported filters for notification. The Event Notification Service supports several filter combinations. Each filter contains a list, which can be empty. When the list contains more than one item, the elements can be separated with semicolons. The filtered values can be defined as either included or excluded to notify the remote OSS clients.

Table 6-1 Supported Notification Filters

Filter Type	**Raw Events Notification Service**	**Event Notification Service**	
		Network Events	**EMS Internal Events**
Source	Supported.	Supported	Supported.
Ticket or Event Filter	Not supported.	Supported	Supported. Tickets are not generated for system, provisioning, and security events.
Event Category	Not supported.	Supported	Supported.
Event Type	Supported The type can be either 1000 (nonactionable trap) or 1001 (nonactionable syslog).	Supported	Supported.
Severity	Supported only for nonactionable syslog.	Supported	Supported.
Ticket Update Filter	Not supported.	Supported	Not supported.

Note Audit events are not included as part of a notification message.

6.1.3 Registering for Event Notification Service

You must install an SNMP listener utility (trap receiver) on your remote client machine. When you use SNMP, your client does not connect directly to the Prime Network gateway. Your client application needs an SNMP trap receiver on which port and destination IP address are available to receive Prime Network traps.

Event Notification Service registration requires the following parameters:

- snmp-community—SNMP community string used for sending the SNMP notifications. The same community string is used for all destinations.

Community string is a password that allows access to a SNMP agent running on the OSS client (Netcool, MIB-Browser, or any other application, which is used to receive traps). SNMP agent can ignore traps that do not contain specific predefined community string or password, and avoid noise, and refer to traps only from the authorized trap generator.

Multiple OSS clients can be registered to the Event Notification Services. The community string defined by clients may vary and it should match the value defined in the SNMP agent. Based on the trap receiver application on the OSS client, the SNMP agent may or may not check the value of the community string.

- snmp-version—SNMP version; for example, v1, v2.
- destination-list—IP addresses and port numbers to which to send SNMP notifications.
- Protocol—Transport protocol; for example, UDP, TCP.

For information on subscribing to receive notifications, see Sample BQL Scripts for Event Notifications, page 358.

6.1.4 Supported User Operations using BQL

Table 6-2 lists the supported notification service user operations that you can perform using the BQL commands.

Table 6-2 Supported BQL for Event Notification Service

Command Name	IMO/OID Type	IMO/OID Value	Description
Create	com.sheer.imo.keys.IOSSClientInfo	{[OssClientInfoRoot] [OSSClientInfo]}	Creates a subscription to receive notifications. The following parameters must be specified: • Destination IP—IP address of the OSS listener. • Port—Listening port details. • Protocol—Transfer protocol UDP or TCP. • Community string—SNMP community string for the generated traps. • SNMP Version—v1 or v2. • Event Notification Service—EventTrapNotification; this value is used for Event Notification Service network events. For the Raw Event Notification Service and for Event Notification Service EMS internal events, use the filter properties (event category) along with EventTrapNotification value.
Update	com.sheer.imo.keys.IOSSClientInfo	{[OssClientInfoRoot] [OSSClientInfo(Id=*Registration ID Number*)]}	Updates the subscription. You can add or remove filters, and change the destination IP address, port, protocol, community string, and SNMP version. The syntax for update is similar to IAddNotification and IScalarNotification. You can use either of these to update the filter.
Get	—	{[OssClientInfoRoo	Retrieves all registration details.

6.1.5 Sample BQL Scripts for Event Notifications

This section contains the following sample BQL scripts:

- Subscribing to Raw Event Notification Service and Event Notification Service for Specific Event Types, page 359
- Subscribing to Raw Event Notification Service and Event Notification Service for All Event Types, page 360Creating **New Subscription by Including Certain Ticket Update Properties**, page 362
- Creating New Subscription Registered on all Managed Network Elements, page 363
- Updating Destination IP Address, page 364
- Updating Community String, page 365
- Adding Event Type Filter, page 365**Adding Source IP Address Filter**, page 366
- Updating Event Type Filter and Source IP Address, page 366
- Adding Severity Filter, page 368

Subscribing to Raw Event Notification Service and Event Notification Service for Specific Event Types

The following example shows the usage of the BQL **Create** command to register for all event categories with all supported severities. The following parameters are used:

- Destination IP address is 192.162.10.18.
- Community string is public.
- UDP protocol is used with port number 162.
- Traps are notified in SNMP v1 format.
- Event type filter is used to generate notification for standard traps (1000) and exclude notification for login (404), logoff (405), and provisioning (998). The severity is Critical (6) and Major (5).

- Receives events notifications from all network elements (IPSubnet is 0.0.0.0,0.0.0.0) that are managed in Prime Network except for 10.77.214.140,255.255.255.255 and 192.168.1.0,255.255.255.255.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Create">
  <param name="imobject">
        <value>
              <IOSSClientInfo>
                <ID
              type="Oid">{[OssClientInfoRoot][OSSClientInfo]}</ID>
              <Address
              type="com.sheer.types.IPAddress">192.162.10.18</Addre
              ss>
              <CommunityString
              type="String">public</CommunityString>
              <ConnectionType
              type="com.sheer.types.sbc.TransportProtocolEnum">UDP<
              /ConnectionType>
              <Description type="String">BQL Event Notification
              Service </Description>
              <Filter type="IOssEventTrapFilter">
                    <ID type="Oid">{[OSSClientFilter]}</ID>
                    <ExcludedEventTypes type="int_Array">
                          <int>404</int>
                          <int>405</int>
                          <int>998</int>
                    </ExcludedEventTypes>
                    <ExcludedSourceIPs
              type="com.sheer.types.IPSubnet_Array">

<com.sheer.types.IPSubnet>10.77.214.140,255.255.255.255</com.sheer.types.I
PSubnet>
```

```
<com.sheer.types.IPSubnet>192.168.1.0,255.255.255.255</com.sheer.types.IPS
ubnet>
                          </ExcludedSourceIPs>
                          <IncludedEventCategories
                        type="com.sheer.types.enums.NotificationCategories
                        Enum_Array">

<com.sheer.types.enums.NotificationCategoriesEnum>Provisioning</com.sheer.t
ypes.enums.Noti ficationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>Security</com.sheer.types
.enums.Notifica tionCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>Ticket</com.sheer.types.e
nums.Notificati onCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>TicketUpdate</com.sheer.t
ypes.enums.Noti ficationCategoriesEnum>
                          </IncludedEventCategories>
                          <IncludedEventTypes type="int_Array">
                                <int>1000</int>
                          </IncludedEventTypes>
                          <IncludedSeverities type="int_Array">
                                <int>5</int>
                                <int>6</int>
                          </IncludedSeverities>
                          <IncludedSourceIPs
                          type="com.sheer.types.IPSubnet_Array">

<com.sheer.types.IPSubnet>0.0.0.0,0.0.0.0</com.sheer.types.IPSubnet>
                          </IncludedSourceIPs>
                        </Filter>
                        <Name type="String">BQL Test  Event Notification
                        Service </Name>
                        <Port type="Integer">162</Port>
                        <SnmpVersion
                        type="com.sheer.types.enums.NotificationSnmpVersio
                        nEnum">SnmpV1</SnmpVersion>
                        <Type
                      type="com.sheer.types.enums.NotificationTypeEnum">Even
                      tTrapNotification</Type>
                      </IOSSClientInfo>
                    </value>
                </param>
          </command>
```

Subscribing to Raw Event Notification Service and Event Notification Service for All Event Types

The following example shows the usage of the BQL **Create** command to register for all event categories and event types. The following parameters are used:

- Destination IP address is 192.162.10.18.

- Community string is public.
- UDP protocol is used with port number 162.
- Traps are notified in SNMP v1 format.
- All event types are included (-1). The severity is Critical (6).
- Receives events notifications only from 10.77.211.216,255.255.255.255.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Create">
      <param name="imobject">
            <value>
                  <IOSSClientInfo>
                        <ID
                        type="Oid">{[OssClientInfoRoot][OSSClientInfo]}<
                        /ID>
                        <Address
                        type="com.sheer.types.IPAddress">192.168.10.18</
                        Address>
                        <CommunityString
                        type="String">public</CommunityString>
                        <ConnectionType
                        type="com.sheer.types.sbc.TransportProtocolEnum"
                        >UDP</ConnectionType>
                        <Description type="String">BQL Event
                        Notification - All Event
  Types</Description>
      <Filter type="IOssEventTrapFilter">
            <ID type="Oid">{[OSSClientFilter]}</ID>
            <ExcludedEventTypes type="int_Array" />
            <ExcludedSourceIPs type="com.sheer.types.IPSubnet_Array" />
            <IncludedEventCategories
            type="com.sheer.types.enums.NotificationCategoriesEnum_Array
            ">
<com.sheer.types.enums.NotificationCategoriesEnum>Provisioning</com.sheer.t
ypes.enums.Noti ficationCategoriesEnum>
            </IncludedEventCategories>
            <IncludedEventTypes type="int_Array">
                  <int>-1</int>
            </IncludedEventTypes>
            <IncludedSeverities type="int_Array">
                  <int>6</int>
            </IncludedSeverities>
            <IncludedSourceIPs type="com.sheer.types.IPSubnet_Array">

<com.sheer.types.IPSubnet>10.77.211.216,255.255.255.255</com.sheer.types.I
PSubnet>
            </IncludedSourceIPs>
          </Filter>
          <Name type="String">BQL Event Notification - All Event
          Types</Name>
          <Port type="Integer">162</Port>
          <SnmpVersion
          type="com.sheer.types.enums.NotificationSnmpVersionEnum">SnmpV
          1</SnmpVersion>
```

```
            <Type
          type="com.sheer.types.enums.NotificationTypeEnum">EventTrapNotific
          ation</Type>
          </IOSSClientInfo>
        </value>
    </param>
</command>
.
```

Creating New Subscription by Including Certain Ticket Update Properties

The following example shows the usage of the BQL **Create** command to create a new subscription with the ticket update properties: AGGREGATEDACKSTATEENUM and DESCRIPTION.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Create">
  <param name="imobject">
        <value>
              <IOSSClientInfo type="IOSSClientInfo" instance_id="0">
                    <ID
                  type="Oid">{[OssClientInfoRoot][OSSClientInfo]}</ID>
                    <Name type="String">test6</Name>
                    <Filter type="IOssEventTrapFilter" instance_id="1">
                        <ID type="Oid">{[OSSClientFilter]}</ID>
                        <IncludedSeverities type="int_Array">
                        <java.lang.Integer>6</java.lang.Integer>
                        </IncludedSeverities>
                        <IncludedSourceIPs
                      type="com.sheer.types.IPSubnet_Array">
                        <com.sheer.types.IPSubnet>0.0.0.0,0.0.0.0</com.she
                      er.types.IPSubnet>
                        </IncludedSourceIPs>
                        <IncludedEventCategories
                      type="com.sheer.types.enums.NotificationCategoriesEn
                      um_Array">
<com.sheer.types.enums.NotificationCategoriesEnum>TicketUpdate</com.sheer.t
ypes.enums.Noti ficationCategoriesEnum>
                         </IncludedEventCategories>
                         <IncludedEventTypes type="int_Array">
                                <java.lang.Integer>-1</java.lang.Integer>
                          </IncludedEventTypes>
                          <ExcludedSourceOids
                        type="com.sheer.framework.imo.Oid_Array" />
                         <AllManagedSourceIpsIncluded
                         type="Boolean">true</AllManagedSourceIpsIncluded>
                         <ExcludedSourceIPs
                         type="com.sheer.types.IPSubnet_Array" />
                         <IncludedSourceOids
                         type="com.sheer.framework.imo.Oid_Array" />
                         <ExcludedEventTypes type="int_Array" />
                         <IncludedTicketProperties
                         type="java.lang.String_Array">
                         <java.lang.String>AggregatedAckStateEnum</java.la
                         ng.String>
                         <java.lang.String>Description</java.lang.String>
                         </IncludedTicketProperties>
                    </Filter>
```

```
                <SnmpVersion
                type="com.sheer.types.enums.NotificationSnmpVersionEnum"
                >SnmpV1</SnmpVersion>
                <Description type="String" />
                <Port type="Integer">162</Port>
                <CommunityString type="String">public</CommunityString>
                <State
                type="com.sheer.types.enums.AdministrativeStateEnum">Up<
                /State>
                <NotificationCount type="Long">0</NotificationCount>
                <Type
                type="com.sheer.types.enums.NotificationTypeEnum">EventT
                rapNotification</Type>
                <Address
                type="com.sheer.types.IPAddress">6.6.6.6</Address>
                <ConnectionType
                type="com.sheer.types.sbc.TransportProtocolEnum">UDP</Co
                nnectionType>
             </IOSSClientInfo>
          </value>
       </param>
     </command>
```

.

Creating New Subscription Registered on all Managed Network Elements

The following example shows the usage of the BQL **Create** command to create a new subscription registered on all managed network elements.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Create">
  <param name="imobject">
    <value>
        <IOSSClientInfo type="IOSSClientInfo" instance_id="0">
           <ID type="Oid">{[OssClientInfoRoot][OSSClientInfo]}</ID>
           <Name type="String">test5</Name>
           <Filter type="IOssEventTrapFilter" instance_id="1">
              <ID type="Oid">{[OSSClientFilter]}</ID>
              <IncludedSeverities type="int_Array">
                    <java.lang.Integer>6</java.lang.Integer>
              </IncludedSeverities>
              <IncludedSourceIPs type="com.sheer.types.IPSubnet_Array" />
              <IncludedEventCategories
        type="com.sheer.types.enums.NotificationCategoriesEnum_Array">
<com.sheer.types.enums.NotificationCategoriesEnum>Ticket</com.sheer.types.e
nums.Notificati onCategoriesEnum>
              </IncludedEventCategories>
              <IncludedEventTypes type="int_Array">
                    <java.lang.Integer>-1</java.lang.Integer>
              </IncludedEventTypes>
              <ExcludedSourceOids type="com.sheer.framework.imo.Oid_Array"
              />
              <AllManagedSourceIpsIncluded
              type="Boolean">true</AllManagedSourceIpsIncluded>
              <ExcludedSourceIPs type="com.sheer.types.IPSubnet_Array" />
              <IncludedSourceOids type="com.sheer.framework.imo.Oid_Array"
              />
              <ExcludedEventTypes type="int_Array" />
```

```
            <IncludedTicketProperties type="java.lang.String_Array" />
            </Filter>
            <SnmpVersion
            type="com.sheer.types.enums.NotificationSnmpVersionEnum">Snm
            pV1</SnmpVersion>
            <Description type="String" />
            <Port type="Integer">162</Port>
            <CommunityString type="String">public</CommunityString>
            <State
            type="com.sheer.types.enums.AdministrativeStateEnum">Up</St
            ate>
            <NotificationCount type="Long">0</NotificationCount>
            <Type
            type="com.sheer.types.enums.NotificationTypeEnum">EventTrapN
            otification</Type>
            <Address type="com.sheer.types.IPAddress">5.5.5.5</Address>
            <ConnectionType
            type="com.sheer.types.sbc.TransportProtocolEnum">UDP</Conne
            ctionType>
         </IOSSClientInfo>
      </value>
   </param>
</command>
.
```

Updating Destination IP Address

The following example shows the usage of the BQL **Update** command to update the existing notification service with destination IP address. Here, the IScalarNotification notification type syntax is used to update the notification.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Update">
  <param name="oid">
        <value>{[OssClientInfoRoot][OSSClientInfo(Id=1)]}</value>
  </param>
  <param name="imobjectArr">
        <value>
              <IScalarNotification>
                    <NewIMO type="IOSSClientInfo">
                       <ID
                       type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id=1)
                       ]}</ID>
                       <Address
                       type="com.sheer.types.IPAddress">192.168.10.19</Add
                       ress>
                    </NewIMO>
                    <OldIMO type="IOSSClientInfo">
                       <ID
                       type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id=1)
                       ]}</ID>
                       <Address
                       type="com.sheer.types.IPAddress">192.168.10.18</Add
                       ress>
                    </OldIMO>
                    <PropertyName type="String">Address</PropertyName>
              </IScalarNotification>
           </value>
        </param>
```

```
  </command>
.
```

Updating Community String

The following example shows the usage of the BQL **Update** command to update the existing notification service with community string. Here, the IScalarNotification notification type syntax is used to update the notification.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Update">
  <param name="oid">
        <value>{[OssClientInfoRoot][OSSClientInfo(Id=1)]}</value>
  </param>
  <param name="imobjectArr">
        <value>
              <IScalarNotification>
                    <NewIMO type="IOSSClientInfo">
                       <ID
                       type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id=1)
                       ]}</ID>
                       <CommunityString
                       type="String">testpublic</CommunityString>
                    </NewIMO>
                    <OldIMO type="IOSSClientInfo">
                       <ID
                       type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id=1)
                       ]}</ID>
                       <CommunityString
                       type="String">public</CommunityString>
                    </OldIMO>
                    <PropertyName
                 type="String">CommunityString</PropertyName>
              </IScalarNotification>
        </value>
  </param>
</command>
.
```

Adding Event Type Filter

The following example shows the usage of the BQL **Update** command to add the event type filter to the existing notification service. Here, the IAddNotification notification type syntax is used to add the event type filter 926 (power supply state change trap) and 1221 (BGP link down).

Refer to either the alarm-types.xml or site.xml file to get the integer values for the event type.

```
<command name="Update">
  <param name="oid">
        <value>{[OssClientInfoRoot][OSSClientInfo]}</value>
  </param>
  <param name="imobjectArr">
        <value>
              <IAddNotification>
                    <NewIMO type="IOSSClientFilter">
```

```
                              <ID
type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id=0)][OSSClientFilter]}</ID
>
                              <IncludedEventTypes type="int_Array">
                                    <int>926</int>
                                    <int>1221</int>
                              </IncludedEventTypes>
                        </NewIMO>
                        <OldIMO type="IOSSClientInfo">
                        </OldIMO>
                        <PropertyName
                  type="String">IncludedEventTypes</PropertyName>
                  </IAddNotification>
         </value>
   </param>
</command>
```

Adding Source IP Address Filter

The following example shows the usage of the BQL **Update** command to add the source IP address filter to the existing notification service. Here, the IAddNotification notification type syntax is used to add the filter.

```
<command name="Update">
   <param name="oid">
         <value>{[OssClientInfoRoot][OSSClientInfo]}</value>
   </param>
   <param name="imobjectArr">
         <value>
               <IAddNotification>
                     <NewIMO type="IOSSClientFilter">
                              <ID
type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id=0)][OSSClientFilter]}</ID
>
                        <IncludedSourceIPs
                     type="com.sheer.types.IPSubnet_Array">
                           <IPSubnet>192.168.10.10,192.168.255.255</IPSubnet>
                           <IPSubnet>172.16.12.2,172.16.255.255</IPSubnet>
                        </IncludedSourceIPs>
                     </NewIMO>
                  <OldIMO type="IOSSClientFilter">
                  </OldIMO>
                  <PropertyName type="String">IncludedSourceIPs</PropertyName>
               </IAddNotification>
         </value>
   </param>
</command>
.
```

Updating Event Type Filter and Source IP Address

The following example shows the usage of the BQL **Update** command to add the event type filter to the existing notification service. Here, the IScalarNotification notification type syntax is used to add the event type filter 555 (mpls te tunnel down).
Refer to either the alarm-types.xml or site.xml file to get the integer values for the event type.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Update">
  <param name="oid">
        <value>{[OssClientInfoRoot][OSSClientInfo(Id=1)]}</value>
  </param>
  <param name="imobjectArr">
        <value>
              <IScalarNotification>
                  <NewIMO type="IOSSClientInfo">
                     <ID
                     type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id=1)]}
                     </ID>
                     <Filter type="IOssEventTrapFilter">
                          <ID type="Oid">{[OSSClientFilter(Id=1)]}</ID>
                          <ExcludedEventTypes type="int_Array" />
                          <ExcludedSourceIPs
                          type="com.sheer.types.IPSubnet_Array" />
                          <IncludedEventCategories
                          type="com.sheer.types.enums.NotificationCategori
                          esEnum_Array">

<com.sheer.types.enums.NotificationCategoriesEnum>Provisioning</com.sheer.t
ypes.enums.Noti ficationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V1Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V2Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V3Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>
                          </IncludedEventCategories>
                          <IncludedEventTypes type="int_Array">
                                <int>555</int>
                          </IncludedEventTypes>
                          <IncludedSeverities type="int_Array">
                                <int>6</int>
                          </IncludedSeverities>
                          <IncludedSourceIPs
                     type="com.sheer.types.IPSubnet_Array">

<com.sheer.types.IPSubnet>10.77.211.216,255.255.255.255</com.sheer.types.I
PSubnet>

<com.sheer.types.IPSubnet>10.77.214.140,255.255.255.255</com.sheer.types.I
PSubnet>
                          </IncludedSourceIPs>
                        </Filter>
                     </NewIMO>
                     <OldIMO type="IOSSClientInfo">
                       <ID
                       type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id=1)
                       ]}</ID>
                       <Filter type="IOssEventTrapFilter">
                          <ID type="Oid">{[OSSClientFilter(Id=1)]}</ID>
                          <ExcludedEventTypes type="int_Array" />
```

```
                              <ExcludedSourceIPs
                              type="com.sheer.types.IPSubnet_Array" />
                              <IncludedEventCategories
                              type="com.sheer.types.enums.NotificationCategori
                              esEnum_Array">

<com.sheer.types.enums.NotificationCategoriesEnum>Provisioning</com.sheer.t
ypes.enums.Noti ficationCategoriesEnum>
                              </IncludedEventCategories>
                              <IncludedEventTypes type="int_Array">
                                    <int>-1</int>
                              </IncludedEventTypes>
                              <IncludedSeverities type="int_Array">
                                    <int>6</int>
                              </IncludedSeverities>
                              <IncludedSourceIPs
                        type="com.sheer.types.IPSubnet_Array">

<com.sheer.types.IPSubnet>10.77.211.216,255.255.255.255</com.sheer.types.I
PSubnet>
                              </IncludedSourceIPs>
                        </Filter>
                     </OldIMO>
                  <PropertyName type="String">Filter</PropertyName>
               </IScalarNotification>
            </value>
   </param>
</command>
.
```

Adding Severity Filter

The following example shows the usage of the BQL **Update** command to add the severity filter (major [5] and minor [4]) to the existing notification service. Here, the IAddNotification notification type syntax is used to add the filter.

```
<command name="Update">
   <param name="oid">
         <value>{[OssClientInfoRoot][OSSClientInfo]}</value>
   </param>
   <param name="imobjectArr">
         <value>
               <IAddNotification>
                     <NewIMO type="IOSSClientFilter">
                              <ID
type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id=0)][OSSClientFilter]}</ID
>
                              <IncludedSeverities type="int_Array">
                                    <int>5</int>
                                    <int>4</int>
                                </IncludedSeverities>
                     </NewIMO>
                     <OldIMO type="IOSSClientFilter">
                     </OldIMO>
                     <PropertyName
               type="String">IncludedSeverities</PropertyName>
               </IAddNotification>
```

```
            </value>
        </param>
</command>
.
```

Updating Severity Filter

The following example shows the usage of the BQL **Update** command to add the severity filter (3-warning, 4-minor, and 5-major) to the existing notification service. Here, the IScalarNotification notification type syntax is used to add the filter.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Update">
  <param name="oid">
        <value>{[OssClientInfoRoot][OSSClientInfo(Id=1)]}</value>
  </param>
  <param name="imobjectArr">
        <value>
              <IScalarNotification>
                    <NewIMO type="IOSSClientInfo">
                          <ID
                          type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id
                          =1)]}</ID>
                          <Filter type="IOssEventTrapFilter">
                            <ID type="Oid">{[OSSClientFilter(Id=1)]}</ID>
                            <ExcludedEventTypes type="int_Array" />
                            <ExcludedSourceIPs
                            type="com.sheer.types.IPSubnet_Array" />
                            <IncludedEventCategories
                            type="com.sheer.types.enums.NotificationCatego
                            riesEnum_Array">
<com.sheer.types.enums.NotificationCategoriesEnum>Provisioning</com.sheer.t
ypes.enums.Noti ficationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V1Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V2Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V3Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>
                            </IncludedEventCategories>
                            <IncludedEventTypes type="int_Array">
                              <int>209</int>
                            </IncludedEventTypes>
                            <IncludedSeverities type="int_Array">
                              <int>3</int>
                              <int>4</int>
                              <int>5</int>
                              <int>6</int>
                            </IncludedSeverities>
                          <IncludedSourceIPs
                    type="com.sheer.types.IPSubnet_Array">

<com.sheer.types.IPSubnet>10.77.211.216,255.255.255.255</com.sheer.types.I
PSubnet>
```

```
<com.sheer.types.IPSubnet>10.77.214.140,255.255.255.255</com.sheer.types.I
PSubnet>
                          </IncludedSourceIPs>
                        </Filter>
                    </NewIMO>
                    <OldIMO type="IOSSClientInfo">
                          <ID
                          type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id
                          =1)]}</ID>
                          <Filter type="IOssEventTrapFilter">
                             <ID type="Oid">{[OSSClientFilter(Id=1)]}</ID>
                             <ExcludedEventTypes type="int_Array" />
                             <ExcludedSourceIPs
                             type="com.sheer.types.IPSubnet_Array" />
                             <IncludedEventCategories
                             type="com.sheer.types.enums.NotificationCateg
                             oriesEnum_Array">

<com.sheer.types.enums.NotificationCategoriesEnum>Provisioning</com.sheer.t
ypes.enums.Noti ficationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V1Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V2Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V3Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>
                                  </IncludedEventCategories>
                                  <IncludedEventTypes type="int_Array">
                                  <int>209</int>
                                  </IncludedEventTypes>
                                  <IncludedSeverities type="int_Array">
                                  <int>6</int>
                                  </IncludedSeverities>
                                  <IncludedSourceIPs
                                  type="com.sheer.types.IPSubnet_Array">

<com.sheer.types.IPSubnet>10.77.211.216,255.255.255.255</com.sheer.types.I
PSubnet>

<com.sheer.types.IPSubnet>10.77.214.140,255.255.255.255</com.sheer.types.I
PSubnet>
                                  </IncludedSourceIPs>
                               </Filter>
                      </OldIMO>
                      <PropertyName type="String">Filter</PropertyName>
                    </IScalarNotification>
              </value>
        </param>
</command>
.
```

Adding Event Category Filter

The following example shows the usage of the BQL **Update** command to add the event category (security and service events) filter to the existing notification service. Here, the IScalarNotification notification type syntax is used to add the filter.

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Update">
  <param name="oid">
        <value>{[OssClientInfoRoot][OSSClientInfo(Id=1)]}</value>
  </param>
  <param name="imobjectArr">
        <value>
              <IScalarNotification>
                  <NewIMO type="IOSSClientInfo">
                    <ID
              type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id=1)]}</ID>
                    <Filter type="IOssEventTrapFilter">
                      <ID type="Oid">{[OSSClientFilter(Id=1)]}</ID>
                      <ExcludedEventTypes type="int_Array" />
                      <ExcludedSourceIPs
                      type="com.sheer.types.IPSubnet_Array" />
                      <IncludedEventCategories
                      type="com.sheer.types.enums.NotificationCategoriesEn
                      um_Array">

<com.sheer.types.enums.NotificationCategoriesEnum>Provisioning</com.sheer.t
ypes.enums.Noti ficationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>Security</com.sheer.types
.enums.Notifica tionCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>Service</com.sheer.types.
enums.NotificationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V1Trap</com.sheer.types.e
nums.NotificationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V2Trap</com.sheer.types.e
nums.NotificationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V3Trap</com.sheer.types.e
nums.NotificationCategoriesEnum>
                      </IncludedEventCategories>
                      <IncludedEventTypes type="int_Array">
                          <int>209</int>
                      </IncludedEventTypes>
                       <IncludedSeverities type="int_Array">
                          <int>3</int>
                          <int>4</int>
                          <int>5</int>
                          <int>6</int>
                      </IncludedSeverities>
                     <IncludedSourceIPs
                  type="com.sheer.types.IPSubnet_Array">

<com.sheer.types.IPSubnet>10.77.211.216,255.255.255.255</com.sheer.types.I
PSubnet>
```

```
<com.sheer.types.IPSubnet>10.77.214.140,255.255.255.255</com.sheer.types.I
PSubnet>
                    </IncludedSourceIPs>
               </Filter>
             </NewIMO>
             <OldIMO type="IOSSClientInfo">
               <ID
               type="Oid">{[OssClientInfoRoot][OSSClientInfo(Id=1)]}</ID>
               <Filter type="IOssEventTrapFilter">
                     <ID type="Oid">{[OSSClientFilter(Id=1)]}</ID>
                     <ExcludedEventTypes type="int_Array" />
                     <ExcludedSourceIPs
                     type="com.sheer.types.IPSubnet_Array" />
                     <IncludedEventCategories
                     type="com.sheer.types.enums.NotificationCategoriesEnu
                     m_Array">

<com.sheer.types.enums.NotificationCategoriesEnum>Provisioning</com.sheer.t
ypes.enums.Noti ficationCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V1Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V2Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>

<com.sheer.types.enums.NotificationCategoriesEnum>V3Trap</com.sheer.types.e
nums.Notificati onCategoriesEnum>
                     </IncludedEventCategories>
                     <IncludedEventTypes type="int_Array">
                           <int>209</int>
                     </IncludedEventTypes>
                     <IncludedSeverities type="int_Array">
                           <int>3</int>
                           <int>4</int>
                           <int>5</int>
                           <int>6</int>
                     </IncludedSeverities>
                     <IncludedSourceIPs
                  type="com.sheer.types.IPSubnet_Array">

<com.sheer.types.IPSubnet>10.77.211.216,255.255.255.255</com.sheer.types.I
PSubnet>

<com.sheer.types.IPSubnet>10.77.214.140,255.255.255.255</com.sheer.types.I
PSubnet>
                     </IncludedSourceIPs>
                   </Filter>
               </OldIMO>
               <PropertyName type="String">Filter</PropertyName>
          </IScalarNotification>
       </value>
     </param>
</command>
.
```

Retrieving All Registration Details

The following example shows the usage of the BQL **Get** command to retrieve all the Event Notification Service registration details.

```
<command name="Get">
  <param name="oid">
        <value>{[OssClientInfoRoot]}</value>
  </param>
  <param name="rs">
        <value><key name="">
      <entry name="depth">100</entry>
      <entry name="register">true</entry>
      <entry name="cachedResultAcceptable">false</entry>
      <key name="requiredProperties">
        <key name="*">
              <entry name="*"/>
        </key>
      </key>
</key></value>
  </param>
</command>
.
```

Retrieving Registration Details for a Specified OID

The following example shows the usage of the BQL **Get** command to retrieve specific Event Notification Service registration details based on registration ID.

```
<command name="Get">
  <param name="oid">
        <value>{[OssClientInfoRoot][OSSClientInfo(Id=1)]}</value>
  </param>
  <param name="rs">
        <value>
              <key name="Test">
                    <entry name="register">false</entry>
                    <key name="requiredProperties">
                          <key name="*">
                                <entry name="*"/>
                          </key>
                    </key>
              </key>
        </value>
  </param>
</command>
.
```

Unsubscribing from Event Notification Service

The following example shows the usage of the BQL **Delete** command to delete the Event Notification Service registration for a specific registration ID.

```
<?xml version="1.0" encoding="UTF-8"?>
  <command name="Delete">
        <param name="oid">
              <value>{[OssClientInfoRoot][OSSClientInfo(Id=1)]}</value>
        </param>
  </command>
.
```

6.2 Understanding the Cisco EPM Notification MIB

When an event is received in the Event Notification Service, it is converted to the trap format defined in the CISCO-EPM-NOTIFICATION-MIB. Other MIB objects are not supported. All OSS clients receive the same traps in same trap format.

The CISCO-EPM-NOTIFICATION-MIB can be found at: http://tools.cisco.com/Support/SNMP/do/BrowseMIB.do?local=en&mibName=CISCO-EPM-NOTIFI CATION-MIB

Table 6-3 describes the mapping between the MIB OIDs to the relevant value that is assigned by the Event Notification Service.

Note All OIDs are suffixed with an Index value. This value is suffixed for every trap and it is incremented for the subsequent trap for every OSS client.

Table 6-3 CISCO EPM-NOTIFICATION-MIB Summary

Trap Field Name	OID	Type	Content as in Trap Forwarder (EPM MIB)
cenAlarmVersion	1.3.6.1.4.1.9.9.311. 1.1.2.1.2.index	SnmpAdmin String	The version of the MIB. The version string will be of the format *major version*.*minor version*. **Note** Always set to 2.2.

Trap Field Name	OID	Type	Content as in Trap Forwarder (EPM MIB)
cenAlarmTimestamp	1.3.6.1.4.1.9.9.311. 1.1.2.1.3.index	Timestamp	The time when the alarm was raised. To find this value, you can use the Java method: • e.getDetectionTime() for all events (security event, system event, provisioning event, syslog, trap, and service alarm). • t.getLastEventTime() for tickets and ticket updates. The cenAlarmTimestamp value is contained in the SNMP TimeTicks Variable Binding type, which represents the time in hundredths of a second. The event creation time (long) value in Cisco Prime Network is divided by 10 and modulo by (2^32)-1 before it is packaged. For example: Prime Network Event Creation time = X cenAlarmTimestamp = (X / 10)%((2^32) - 1)
cenAlarmUpdated Time stamp	1.3.6.1.4.1.9.9.311. 1.1.2.1.4.index	Timestamp	Alarms persist over time and their fields can change values. The last time a field changed, this alarm was updated. The updated time denotes this time. To find this value, you can use the Java method: • e.getDetectionTime() for all events (security event, system event, provisioning event, syslog, trap, and service alarm). • t.getLastEventTime() for ticket. • t.getLastModificationTime() for ticket updates.
cenAlarmInstanceID	1.3.6.1.4.1.9.9.311. 1.1.2.1.5.index	SnmpAdmin String	The unique alarm instance ID. To find this value, you can use this Java method: (Oid)t.getObjectId().
cenAlarmStatus	1.3.6.1.4.1.9.9.311. 1.1.2.1.6.index	Integer32	Event type as defined in the alarm-types.xml file.

Trap Field Name	OID	Type	Content as in Trap Forwarder (EPM MIB)
cenAlarmStatus Definition	1.3.6.1.4.1.9.9.311. 1.1.2.1.7.index	SnmpAdmin String	A short description of the type of the alarm. The event type string representation, translated from the alarm-types.xml file using the format (integer, String).
cenAlarmType	1.3.6.1.4.1.9.9.311. 1.1.2.1.8.index	Integer	The event type - unknown (1), direct (2), indirect (3), mixed (4). The possible values are: • 4–Security event, System event, Provisioning event, Syslog (actionable), Trap (actionable), Service alarm, ticket, and ticket update. • 2–Nonactionable (standard) syslog and trap.
cenAlarmCategory	1.3.6.1.4.1.9.9.311. 1.1.2.1.9.index	Integer32	The alarm category. It is represented as an integer value: • 1–Security event • 2–Service alarm • 3–Syslog (actionable and nonactionable) • 4–System event • 5–SNMPv1 trap (actionable and nonactionable) • 6–SNMPv2 and SNMPv3 trap (actionable and nonactionable) • 7–Provisioning event • 8–Ticket • 9–Ticket update The category descriptions are itemized in the cenAlarmCategoryDefinition field (OID.10).

Trap Field Name	OID	Type	Content as in Trap Forwarder (EPM MIB)
cenAlarmCategory Definition	1.3.6.1.4.1.9.9.311. 1.1.2.1.10.index	SnmpAdmin String	This is a string representation of AlarmCategory (number,description): • 1,Security event • 2,Service alarm • 3,Syslog (actionable and nonactionable) • 4,System event • 5,SNMPv1 trap (actionable and nonactionable) • 6,SNMPv2 and SNMPv3 trap (actionable and nonactionable) • 7,Provisioning event • 8,Ticket • 9,Ticket Update
cenAlarmServer AddressType	1.3.6.1.4.1.9.9.311. 1.1.2.1.11.index	InetAddress Type	The type of internet address at which the server that is generating this trap is reachable. This value is set to 1 for IPV4 management.
cenAlarmServer Address	1.3.6.1.4.1.9.9.311. 1.1.2.1.12.index	InetAddress	Prime Network gateway IP address or DNS name.
cenAlarmManaged ObjectClass	1.3.6.1.4.1.9.9.311. 1.1.2.1.13.index	SnmpAdmin String	• For service alarm, syslog, trap, ticket and ticket update, the value is the string OID of (Oid)e.getSource(). • For security event, system event, and provisioning event, this is an empty string ("").
cenAlarmManaged ObjectAddressType	1.3.6.1.4.1.9.9.311. 1.1.2.1.14.index	InetAddress Type	The type of internet address at which the managed object is reachable. This value is set to 1 for IPV4 management.

Trap Field Name	OID	Type	Content as in Trap Forwarder (EPM MIB)
cenAlarmManaged ObjectAddress	1.3.6.1.4.1.9.9.311. 1.1.2.1.15.index	InetAddress	IP Address or DNS name of the managed object: • Security event—Client IP address. • Provisioning event—Prime Network gateway IP address. • System event—Prime Network gateway IP address. • Service alarm—IP address of the network element (event source) or Prime Network gateway, when the event does not refer to a single managed IP address (for example, service event for link is related to two IP addresses.) • Syslog (actionable and nonactionable)—IP address of the network element (event source). • Trap (actionable and nonactionable)—IP address of the network element (event source). • Ticket and Ticket Update—IP address of the root cause network element (event source).
cenAlarmDescription	1.3.6.1.4.1.9.9.311. 1.1.2.1.16.index	OctetString	A detailed description of the alarm. The following Java methods are used: • Security event—e.getDescription() • Provisioning event—e.getStatusEnum() • System event—e.getDescription() • Service alarm—e.getDescription() • Syslog (actionable and nonactionable)—e.getDescription() • Trap (actionable and nonactionable)—e.getDescription() • Ticket—t.getLatestState() • For ticket updates, the updated property is in the format *property-name=value*.

Trap Field Name	OID	Type	Content as in Trap Forwarder (EPM MIB)
cenAlarmSeverity	1.3.6.1.4.1.9.9.311. 1.1.2.1.17.index	Integer32	Indicates the severity of the alarm using an integer value. The valid integers are 0 - 6 (see cenAlarmSeverity Definition). For nonactionable (standard) syslog and trap, the value is set to zero. For ticket and ticket updates, the value is set to the highest severity value of all events under the ticket (as enumeration). The java method t.getAggregatedSeverityEnum() is used.
cenAlarmSeverity Definition	1.3.6.1.4.1.9.9.311. 1.1.2.1.18.index	OctetString	Alarm severity string representation (number,description): • 0,Indeterminate • 1,Information • 2,Cleared • 3,Warning • 4,Minor • 5,Major • 6,Critical
cenAlarmTriageValue	1.3.6.1.4.1.9.9.311. 1.1.2.1.19.index	Integer32	The triage value of an alarm. The value is zero, which denotes an undetermined or uncomputable value.
cenEventIDList	1.3.6.1.4.1.9.9.311. 1.1.2.1.20.index	OctetString	List of event IDs that led to this alarm. Set to zero length string ("").
cenUserMessage1	1.3.6.1.4.1.9.9.311. 1.1.2.1.21.index	SnmpAdmin String	User input message. For nonactionable syslog and trap, the value is set to empty string. See Customizing the cenUserMessage Values, page 381 to customize this field.
cenUserMessage2	1.3.6.1.4.1.9.9.311. 1.1.2.1.22.index	SnmpAdmin String	User input message. For nonactionable syslog and trap, the value is set to empty string. See Customizing the cenUserMessage Values, page 381 to customize this field.

Trap Field Name	OID	Type	Content as in Trap Forwarder (EPM MIB)
cenUserMessage3	1.3.6.1.4.1.9.9.311. 1.1.2.1.23.index	SnmpAdmin String	User input message. For nonactionable syslog and trap, the value is set to empty string. See Customizing the cenUserMessage Values, page 381 to customize this field.
cenAlarmMode	1.3.6.1.4.1.9.9.311. 1.1.2.1.24.index	Integer	The alarm mode. The possible values are: • 1—Provision event, security event, and system event. • 2—Ticket and ticket update. • 3—Service alarm, syslog, and trap (SNMPv1 and SNMPv2).
cenPartitionNumber	1.3.6.1.4.1.9.9.311. 1.1.2.1.25.index	Unsigned32	ID of the logical group of managed devices. Set to zero.
cenPartitionName	1.3.6.1.4.1.9.9.311. 1.1.2.1.26.index	SnmpAdmin String	Name of the logical group of managed devices. Set to zero length string ("").
cenCustomer Identification	1.3.6.1.4.1.9.9.311. 1.1.2.1.27.index	SnmpAdmin String	User input message. For nonactionable syslog and trap, the value is set to empty string. See Customizing the cenUserMessage Values, page 381 to customize this field.
cenCustomerRevision	1.3.6.1.4.1.9.9.311. 1.1.2.1.28.index	SnmpAdmin String	User input message. For nonactionable syslog and trap, the value is set to empty string. See Customizing the cenUserMessage Values, page 381 to customize this field.
cenAlertID	1.3.6.1.4.1.9.9.311. 1.1.2.1.29.index	SnmpAdmin String	Alert ID to which the event is rolled up. For Provision event, Security event, System event, Ticket, and Ticket Update, this field is empty (zero-length string – ""). For Service alarm, syslog (actionable), and trap (actionable) the value is taken from Java method e.getTicketOid().

6.2.1 Customizing the cenUserMessage Values

cenUserMessage1, cenUserMessage2, cenUserMessage3, cenCustomerIdentification and cenCustomerRevision are standard EPM-MIB fields, where users can change the values. The values for these fields can be changed in the avm11.xml file for event and ticket notification. These values are mapped under the key name cenUserMessageX in the avm11.xml file ($*NETWORKHOME*/ Main/registry/ConfigurationFiles/127.0.0.1/) where cenUserMessage4 is mapped to cenCustomerIdentification, and cenUserMessage5 is mapped to cenCustomerRevision.
The key name, cenUserMessageX, is configured to ticketsCenUserMessageMapping for tickets and eventsCenUserMessageMapping for events. The cenUserMessage values can be configured:

- static—The static string is shown in the cenUserMessageX field. In the given avm11.xml example, the value is fixed to Prime Network.
- original—The IMO value of the defined field is shown in the cenUserMessageX field. In the given avm11.xml example, the value is Description, which is the property name of the IMO method imo.getDescription().
- translated—The IMO value of the defined field is shown in the cenUserMessageX field. A user-friendly value is displayed after the translation and the same value may appear in the Prime Network GUI. In the given avm11.xml example, the value is Source, which is the property name of the IMO method imo.getSource().
- NE data: The IMO source's business tag and/or interface description, if they meet the following criteria:
  - Business tags can be included if the tag is defined on the root cause source
  - Interface description can be included if the event or ticket source is an IP interface, and a description was set to the interface

This customization is a system-wide configuration and is applicable to all OSS client subscriptions.
It is also possible to configure this customization on a specific OSS client, ensuring only that client would be affected. This is done from Prime Network Administrator as detailed in Prime Network Administrator Guide.

Editing the avm11.xml Registry File
To change the values for cenUserMessage<X>, you must edit the avm11.xml registry file.

Note Changes to the registry should only be carried out with the support of Cisco. For details, contact your Cisco account representative.

1. Log into the Prime Network gateway as *network-user* (where *network-user* is the operating system user for the Prime Network application, created when Prime Network is installed; an example of *network-user* is network310).
2. Change to the Main directory:
   ```
   # cd $ANAHOME/Main/
   ```
   Where $*ANAHOME* is the Prime Network installation directory.
3. Run the following commands to edit the registry file:

a. Add ticketsCenUserMessageMapping for tickets:
```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping
```
b. Change the cenUserMessage1 value for tickets:
```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUser
Message1

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUs
erMessage1/translated Source
```
Where, *Source* is the IMO value for the defined field (translated). This value is displayed after the translation. See Customizing the cenUserMessage Values, page 381 for more information.

c. Change the cenUserMessage2 value for tickets:
```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUser
Message2

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenU
serMessage2/static Prime Network
```
Where, *Prime Network* is a static string. See Customizing the cenUserMessage Values, page 381 for more information.

d. Change the cenUserMessage3 value for tickets:
```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage3

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage3/original Description
```
Where, Description is the IMO value for the defined field (Original). See Customizing the cenUserMessage Values, page 381 for more information.

e. Change the cenUserMessage4 value for tickets:
```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage4

./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage4/source-data

./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage4/source-data/aspect-property

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage4/source-data/aspect-property/enabled true
```

This will include the business tag information in the trap if such a tag was attached to the ticket's source

f. Change the cenUserMessage5 value for tickets:

```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage5

./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage5/source-data

./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage5/source-data/simple-property

./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage5/source-data/simple-
property/com.sheer.imo.technologies.IIPInterface

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage5/source-data/simple-
property/com.sheer.imo.technologies.IIPInterface/OidClass
com.sheer.imo.keys.IIpInterfaceOid

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/ticketsCenUserMessageMapping/cenUse
rMessage5/source-data/simple-
property/com.sheer.imo.technologies.IIPInterface/portDescription
" "
```

This will include the IP interface description information in the trap if the event's source is an IP interface

g. Add eventsCenUserMessageMapping for events:

```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping
```

h. Change the cenUserMessage1 value for events:

```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message1

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message1/translated Source
```

Where, Source is the IMO value for the defined field (translated). This value is displayed after the translation. For more information , see Customizing the cenUserMessage Values, page 381.

i. Change the cenUserMessage2 value for events:

```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message2

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message2/static Prime Network
```

Where, ANA is a static string. For more information, see Customizing the cenUserMessage Values, page 381.

j. Change the cenUserMessage3 value for events:

```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message3

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message3/original Description
```

Where, Description is the IMO value for the defined field (Original). See Customizing the cenUserMessage Values, page 381 for more information.

k. Change the cenUserMessage4 value for events:

```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message4

./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message4/source-data

./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message4/source-data/aspect-property

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message4/source-data/aspect-property/enabled true
```

This will include the business tag information in the trap if such a tag was attached to the event's source.

l. Change the cenUserMessage5 value for events:

```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message5
```

```
./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message5/source-data

./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message5/source-data/simple-property

./runRegTool.sh -gs 127.0.0.1 add 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message5/source-data/simple-
property/com.sheer.imo.technologies.IIPInterface

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message5/source-data/simple-
property/com.sheer.imo.technologies.IIPInterface/OidClass
com.sheer.imo.keys.IIpInterfaceOid

./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
avm11/services/trapforwarding/eventsCenUserMessageMapping/cenUser
Message5/source-data/simple-
property/com.sheer.imo.technologies.IIPInterface/portDescription
" "
```

This will include the IP interface description information in the trap if the event's source is an IP interface

4. Restart the Prime Network gateway using the command:
   `anactl restart`

Sample avm11.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<key name="avm11">
      <entry name="default">mmvm</entry>
      <key name="services">
            <key name="logger">
                  <entry name="log4j.category.com.sheer">ERROR</entry>
            </key>
            <key name="trapforwarding">
                  <key name="ticketsCenUserMessageMapping">
                        <key name="cenUserMessage1">
                              <entry name="translated">Source</entry>
                        </key>
                        <key name="cenUserMessage3">
                              <entry name="static">Prime
                        Network</entry>
                        </key>
                        <key name="cenUserMessage2">
                              <entry
                        name="original">LastModificationTime</entry>
                        </key>
                        <key name="cenUserMessage5">
```

```
        <key name="source-data">
          <key name="simple-property">
             <key
name="com.sheer.imo.technologies.IIPInterface">

<entry
name="OidClass">com.sheer.imo.keys.IIpInterfaceOid</e
ntry>

<entry name="portDescription"></entry>

      </key>
      </key>
      </key>
      </key>
</key>
<key name="eventsCenUserMessageMapping">
      <key name="cenUserMessage1">
            <entry name="translated">Source</entry>
      </key>
      <key name="cenUserMessage3">
            <entry name="static">Prime
      Network</entry>
      </key>
      <key name="cenUserMessage2">
            <entry
      name="original">DetectionTime</entry>

      </key>
      <key name="cenUserMessage5">
            <key name="source-data">
                  <key name="aspect-property">
                      <entry
                      name="enabled">true</entry>
                  </key>
            </key>
          </key>
          </key>
```

Example 1: Generated SNMP Trap for a SNMPv1 Trap with Customized Value
The following example displays the customized value for cenUserMessage1 (1.3.6.1.4.1.9.9.311.1.1.2.1.21), cenUserMessage2 (1.3.6.1.4.1.9.9.311.1.1.2.1.22), and cenUserMessage3 (1.3.6.1.4.1.9.9.311.1.1.2.1.23).

<VariableBinding OID="1.3.6.1.4.1.9.9.311.1.1.2.1.20.11" Value=""
ValueType="OctetString"/>
<VariableBinding OID="1.3.6.1.4.1.9.9.311.1.1.2.1.21.11" Value="Avm 100"
ValueType="OctetString"/>
<VariableBinding OID="1.3.6.1.4.1.9.9.311.1.1.2.1.22.11" Value="AVM 10.56.58.176:100
(100)
was disabled" ValueType="OctetString"/>

<VariableBinding OID="1.3.6.1.4.1.9.9.311.1.1.2.1.23.11" Value="ANA"
ValueType="OctetString"/>
<VariableBinding OID="1.3.6.1.4.1.9.9.311.1.1.2.1.24.11" Value="1"
ValueType="Integer32"/>

6.3 Event Notification Service Errors and Exceptions

The following types of exceptions may occur when notification messages are being generated:

- TfsConfigurationException—Occurs when there is an issue with the configuration; that is, with Event Notification Service initialization or command handling.
- TfsRuntimeException—Occurs when there is an issue while translating the event or ticket to the SNMP notification; for example, unexpected data in the event or an EPM conversion error.

Changing the Debug Level

You can use the following command to debug the Event Notification Service component:

Note Changes to the registry should only be carried out with the support of Cisco. For details, contact your Cisco account representative.

1. Log into the Cisco Prime Network gateway as network-user (where network-user is the operating system user for the Prime Network application, created when Prime Network is installed; an example of network-user is network39).
2. Change to the Main directory:
   ```
   # cd $ANAHOME/Main/
   ```
   Where $ANAHOME is the Prime Network installation directory.
3. Run the following commands:
   ```
   ./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
   ./runRegTool.sh -gs 127.0.0.1 set 127.0.0.1
   avm11/services/logger/log4j.category.com.sheer.metromission.trapforward
   ing DEBUG
   ```
4. Restart the Prime Network gateway using the following command
   ```
   anactl restart
   ```

6.4 Sample SNMP Notification Examples

The following are some of the examples of the SNMP notification. You must refer to the Table 6-3 for information on mapping between the MIB OIDs to the relevant value that is assigned by the Event Notification Service.

SNMP Notification for Nonactionable Syslog

The following example shows the SNMP notification generated for a nonactionable (standard) syslog (event type 1001; event ID 239075246311_1260947952869) for the managed element (169.254.192.231) with the severity 0. This syslog was generated when the interface aaa changed its state to administratively down (see 1.3.6.1.4.1.9.9.311.1.1.2.1.16). The Cisco Prime Network gateway address is 10.56.56.117 (see 1.3.6.1.4.1.9.9.311.1.1.2.1.12).

```
[V1TRAP[reqestID=0,timestamp=6 days,
15:44:53.85,enterprise=1.3.6.1.4.1.9.9.311,genericTrap=6,specificTrap=2,
VBS[1.3.6.1.4.1.9.9.311.1.1.2.1.2.1602 = 2.2
1.3.6.1.4.1.9.9.311.1.1.2.1.3.1602 = 16:00:29.62
1.3.6.1.4.1.9.9.311.1.1.2.1.4.1602 = 16:00:29.62
1.3.6.1.4.1.9.9.311.1.1.2.1.5.1602 = 239075246311_1260947952869
1.3.6.1.4.1.9.9.311.1.1.2.1.6.1602 = 1001
1.3.6.1.4.1.9.9.311.1.1.2.1.7.1602 = 1001,standard syslog
1.3.6.1.4.1.9.9.311.1.1.2.1.8.1602 = 4
1.3.6.1.4.1.9.9.311.1.1.2.1.9.1602 = 3
1.3.6.1.4.1.9.9.311.1.1.2.1.10.1602 = 3,Syslog
1.3.6.1.4.1.9.9.311.1.1.2.1.11.1602 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.12.1602 = 10.56.56.117
1.3.6.1.4.1.9.9.311.1.1.2.1.13.1602 =
{[ManagedElement(Key=169.254.192.231)][Syslog]}
1.3.6.1.4.1.9.9.311.1.1.2.1.14.1602 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.15.1602 = 169.254.192.231
1.3.6.1.4.1.9.9.311.1.1.2.1.16.1602 = %CHeetah-5-CHANGED: Interface aaaa,
changed state to administratively down
1.3.6.1.4.1.9.9.311.1.1.2.1.17.1602 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.18.1602 = 0,Indeterminate
1.3.6.1.4.1.9.9.311.1.1.2.1.19.1602 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.20.1602 =
1.3.6.1.4.1.9.9.311.1.1.2.1.21.1602 =
1.3.6.1.4.1.9.9.311.1.1.2.1.22.1602 =
1.3.6.1.4.1.9.9.311.1.1.2.1.23.1602 =
1.3.6.1.4.1.9.9.311.1.1.2.1.24.1602 = 3
1.3.6.1.4.1.9.9.311.1.1.2.1.25.1602 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.26.1602 =
1.3.6.1.4.1.9.9.311.1.1.2.1.27.1602 =
1.3.6.1.4.1.9.9.311.1.1.2.1.28.1602 =
1.3.6.1.4.1.9.9.311.1.1.2.1.29.1602 = ]]]
```

SNMP Notification for Nonactionable Trap

The following example shows the SNMP notification generated for a nonactionable (standard) trap (event type 1001; event ID 239075246311_1264327429023) for the managed element (169.254.192.231) with the severity 0.

```
.1.3.6.1.4.1.9.9.311.1.1.2.1.2.1621 --> 2.2
.1.3.6.1.4.1.9.9.311.1.1.2.1.3.1621 --> 3 days, 0 hours, 18 minutes, 57
seconds.
.1.3.6.1.4.1.9.9.311.1.1.2.1.4.1621 --> 3 days, 0 hours, 18 minutes, 57
seconds.
.1.3.6.1.4.1.9.9.311.1.1.2.1.5.1621 --> 239075246311_1264327429023
.1.3.6.1.4.1.9.9.311.1.1.2.1.6.1621 --> 1000
.1.3.6.1.4.1.9.9.311.1.1.2.1.7.1621 --> 1000,standard trap
.1.3.6.1.4.1.9.9.311.1.1.2.1.8.1621 --> 4
.1.3.6.1.4.1.9.9.311.1.1.2.1.9.1621 --> 5
.1.3.6.1.4.1.9.9.311.1.1.2.1.10.1621 --> 5,V1 Trap
.1.3.6.1.4.1.9.9.311.1.1.2.1.11.1621 --> 1
.1.3.6.1.4.1.9.9.311.1.1.2.1.12.1621 --> 10.56.57.194
.1.3.6.1.4.1.9.9.311.1.1.2.1.13.1621 -->
{[ManagedElement(Key=169.254.192.231)][Trap]}
.1.3.6.1.4.1.9.9.311.1.1.2.1.14.1621 --> 1
.1.3.6.1.4.1.9.9.311.1.1.2.1.15.1621 --> 169.254.192.231
.1.3.6.1.4.1.9.9.311.1.1.2.1.16.1621 --> PDU String request id = 0
.1.3.6.1.2.1.2.2.1.1.3 --> 3
.1.3.6.1.2.1.2.2.1.2.3 --> aaa
.1.3.6.1.2.1.2.2.1.3.3 --> 6
.1.3.6.1.4.1.9.2.2.1.1.20.3 --> administratively down
.1.3.6.1.4.1.9.9.311.1.1.2.1.17.1621  -->   0
.1.3.6.1.4.1.9.9.311.1.1.2.1.18.1621  -->   0,Indeterminate
.1.3.6.1.4.1.9.9.311.1.1.2.1.19.1621  -->   0
.1.3.6.1.4.1.9.9.311.1.1.2.1.20.1621  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.21.1621  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.22.1621  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.23.1621  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.24.1621  -->   3
.1.3.6.1.4.1.9.9.311.1.1.2.1.25.1621  -->   0
.1.3.6.1.4.1.9.9.311.1.1.2.1.26.1621  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.27.1621  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.28.1621  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.29.1621  -->
```

SNMP Notification for SNMPv1 Trap

The following example shows the SNMP notification generated for an SNMPv1 trap (event type 1268; network event ID 269140017383_1264323280488) for the managed element (169.254.192.231 Ethernet0/2) with the severity 5. For the trap description, see .1.3.6.1.4.1.9.9.311.1.1.2.1.16.

```
.1.3.6.1.2.1.1.3.0 --> 29 days, 15 hours, 47 minutes, 13 seconds.
.1.3.6.1.6.3.1.1.4.1.0 --> .1.3.6.1.4.1.9.9.311.0.2
.1.3.6.1.4.1.9.9.311.1.1.2.1.2.1622 --> 2.2
.1.3.6.1.4.1.9.9.311.1.1.2.1.3.1622 --> 2 days, 23 hours, 9 minutes, 48
seconds.
.1.3.6.1.4.1.9.9.311.1.1.2.1.4.1622 --> 2 days, 23 hours, 9 minutes, 48
seconds.
```

```
.1.3.6.1.4.1.9.9.311.1.1.2.1.5.1622 -->
{[NetworkEvent(Id=269140017383_1264323280488)]}
.1.3.6.1.4.1.9.9.311.1.1.2.1.6.1622 --> 1268
.1.3.6.1.4.1.9.9.311.1.1.2.1.7.1622 --> 1268,snmp link down
.1.3.6.1.4.1.9.9.311.1.1.2.1.8.1622 --> 4
.1.3.6.1.4.1.9.9.311.1.1.2.1.9.1622 --> 5
.1.3.6.1.4.1.9.9.311.1.1.2.1.10.1622 --> 5,V1 Trap
.1.3.6.1.4.1.9.9.311.1.1.2.1.11.1622 --> 1
.1.3.6.1.4.1.9.9.311.1.1.2.1.12.1622 --> 10.56.57.194
.1.3.6.1.4.1.9.9.311.1.1.2.1.13.1622 -->
{[ManagedElement(Key=169.254.192.231)][PhysicalRoot][Chassis][Slot(SlotNum
=0)][Module][Por t(PortNumber=Ethernet0/2)][PhysicalLayer][Trap]}
.1.3.6.1.4.1.9.9.311.1.1.2.1.14.1622 --> 1
.1.3.6.1.4.1.9.9.311.1.1.2.1.15.1622 --> 169.254.192.231
.1.3.6.1.4.1.9.9.311.1.1.2.1.16.1622 -->
.iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifIndex.3-->3
.iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifDescr.3--
>Ethernet0/1
.iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifType.3-->6
.iso.org.dod.internet.private.enterprises.cisco.local.linterfaces.lifTable
.lifEntry.locIfR
eason.3-->administratively down

.1.3.6.1.4.1.9.9.311.1.1.2.1.17.1622  -->   5
.1.3.6.1.4.1.9.9.311.1.1.2.1.18.1622  -->   5,Major
.1.3.6.1.4.1.9.9.311.1.1.2.1.19.1622  -->   0
.1.3.6.1.4.1.9.9.311.1.1.2.1.20.1622  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.21.1622  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.22.1622  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.23.1622  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.24.1622  -->   3
.1.3.6.1.4.1.9.9.311.1.1.2.1.25.1622  -->   0
.1.3.6.1.4.1.9.9.311.1.1.2.1.26.1622  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.27.1622  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.28.1622  -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.29.1622  -->
```

SNMP Notification for Syslog

The following example shows the SNMP notification generated for a syslog (event type 178; network event ID 226190344423_1260876153865) for the managed element (169.254.192.231) with the severity 1.

```
[V1TRAP[reqestID=0,timestamp=1 day,
14:57:48.18,enterprise=1.3.6.1.4.1.9.9.311,genericTrap=6,specificTrap=2,
VBS[1.3.6.1.4.1.9.9.311.1.1.2.1.2.1601 = 2.2
1.3.6.1.4.1.9.9.311.1.1.2.1.3.1601 = 19:28:13.84
1.3.6.1.4.1.9.9.311.1.1.2.1.4.1601 = 19:28:13.84
1.3.6.1.4.1.9.9.311.1.1.2.1.5.1601 =
{[NetworkEvent(Id=226190344423_1260876153865)]}
1.3.6.1.4.1.9.9.311.1.1.2.1.6.1601 = 178
1.3.6.1.4.1.9.9.311.1.1.2.1.7.1601 = 178,system restart syslog
1.3.6.1.4.1.9.9.311.1.1.2.1.8.1601 = 4
1.3.6.1.4.1.9.9.311.1.1.2.1.9.1601 = 3
1.3.6.1.4.1.9.9.311.1.1.2.1.10.1601 = 3,Syslog
1.3.6.1.4.1.9.9.311.1.1.2.1.11.1601 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.12.1601 = 10.56.56.117
```

```
1.3.6.1.4.1.9.9.311.1.1.2.1.13.1601 =
{[ManagedElement(Key=169.254.192.231)][Syslog]}
1.3.6.1.4.1.9.9.311.1.1.2.1.14.1601 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.15.1601 = 169.254.192.231
1.3.6.1.4.1.9.9.311.1.1.2.1.16.1601 = %SYS-5-RESTART: System restarted --
1.3.6.1.4.1.9.9.311.1.1.2.1.17.1601 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.18.1601 = 1,Information
1.3.6.1.4.1.9.9.311.1.1.2.1.19.1601 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.20.1601 =
1.3.6.1.4.1.9.9.311.1.1.2.1.21.1601 =
1.3.6.1.4.1.9.9.311.1.1.2.1.22.1601 =
1.3.6.1.4.1.9.9.311.1.1.2.1.23.1601 =
1.3.6.1.4.1.9.9.311.1.1.2.1.24.1601 = 3
1.3.6.1.4.1.9.9.311.1.1.2.1.25.1601 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.26.1601 =
1.3.6.1.4.1.9.9.311.1.1.2.1.27.1601 =
1.3.6.1.4.1.9.9.311.1.1.2.1.28.1601 =
1.3.6.1.4.1.9.9.311.1.1.2.1.29.1601 = ]]]
```

SNMP Notification for a Ticket

The following example shows the SNMP notification generated for a ticket (event type 17; ticket ID 6981) for the managed element (169.254.192.8) with the severity 5. This ticket was generated when the CPU was over utilized.

```
[V1TRAP[reqestID=0,timestamp=1 day,
15:36:20.39,enterprise=1.3.6.1.4.1.9.9.311,genericTrap=6,specificTrap=2,
VBS[1.3.6.1.4.1.9.9.311.1.1.2.1.2.1604 = 2.2
1.3.6.1.4.1.9.9.311.1.1.2.1.3.1604 = 177 days, 11:57:48.97
1.3.6.1.4.1.9.9.311.1.1.2.1.4.1604 = 177 days, 11:57:48.97
1.3.6.1.4.1.9.9.311.1.1.2.1.5.1604 = {[NewAlarm(Id=6981)]}
1.3.6.1.4.1.9.9.311.1.1.2.1.6.1604 = 17
1.3.6.1.4.1.9.9.311.1.1.2.1.7.1604 = 17,cpu utilization
1.3.6.1.4.1.9.9.311.1.1.2.1.8.1604 = 4
1.3.6.1.4.1.9.9.311.1.1.2.1.9.1604 = 8
1.3.6.1.4.1.9.9.311.1.1.2.1.10.1604 = 8,Ticket
1.3.6.1.4.1.9.9.311.1.1.2.1.11.1604 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.12.1604 = 10.56.56.117
1.3.6.1.4.1.9.9.311.1.1.2.1.13.1604 =
{[ManagedElement(Key=169.254.192.8)]}
1.3.6.1.4.1.9.9.311.1.1.2.1.14.1604 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.15.1604 = 169.254.192.8
1.3.6.1.4.1.9.9.311.1.1.2.1.16.1604 = CPU over utilized
1.3.6.1.4.1.9.9.311.1.1.2.1.17.1604 = 5
1.3.6.1.4.1.9.9.311.1.1.2.1.18.1604 = 5,Major
1.3.6.1.4.1.9.9.311.1.1.2.1.19.1604 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.20.1604 =
1.3.6.1.4.1.9.9.311.1.1.2.1.21.1604 =
1.3.6.1.4.1.9.9.311.1.1.2.1.22.1604 =
1.3.6.1.4.1.9.9.311.1.1.2.1.23.1604 =
1.3.6.1.4.1.9.9.311.1.1.2.1.24.1604 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.25.1604 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.26.1604 =
1.3.6.1.4.1.9.9.311.1.1.2.1.27.1604 =
1.3.6.1.4.1.9.9.311.1.1.2.1.28.1604 =
```

```
1.3.6.1.4.1.9.9.311.1.1.2.1.29.1604 = ]]]
```

SNMP Notification for Ticket Update

The following example shows the SNMP notification generated for a ticket update (event type 17; ticket ID 7040) for the managed element (169.254.192.214) with the severity 5. This ticket update happened when a note was added to the ticket.

```
[V1TRAP[reqestID=0,timestamp=6 days,
19:13:49.24,enterprise=1.3.6.1.4.1.9.9.311,genericTrap=6,specificTrap=2,
VBS[1.3.6.1.4.1.9.9.311.1.1.2.1.2.1605 = 2.2

1.3.6.1.4.1.9.9.311.1.1.2.1.3.1605 = 178 days, 8:10:43.84
1.3.6.1.4.1.9.9.311.1.1.2.1.4.1605 = 178 days, 8:10:43.84
1.3.6.1.4.1.9.9.311.1.1.2.1.5.1605 = {[NewAlarm(Id=7040)]}
1.3.6.1.4.1.9.9.311.1.1.2.1.6.1605 = 17
1.3.6.1.4.1.9.9.311.1.1.2.1.7.1605 = 17,cpu utilization
1.3.6.1.4.1.9.9.311.1.1.2.1.8.1605 = 4
1.3.6.1.4.1.9.9.311.1.1.2.1.9.1605 = 9
1.3.6.1.4.1.9.9.311.1.1.2.1.10.1605 = 9,Ticket Update
1.3.6.1.4.1.9.9.311.1.1.2.1.11.1605 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.12.1605 = 10.56.56.117
1.3.6.1.4.1.9.9.311.1.1.2.1.13.1605 =
{[ManagedElement(Key=169.254.192.214)]}
1.3.6.1.4.1.9.9.311.1.1.2.1.14.1605 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.15.1605 = 169.254.192.214
1.3.6.1.4.1.9.9.311.1.1.2.1.16.1605 = Note=--------------- Added By root
on 16-Dec-09
09:40:06 ---------------My New Notes
1.3.6.1.4.1.9.9.311.1.1.2.1.17.1605 = 5
1.3.6.1.4.1.9.9.311.1.1.2.1.18.1605 = 5,Major
1.3.6.1.4.1.9.9.311.1.1.2.1.19.1605 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.20.1605 =
1.3.6.1.4.1.9.9.311.1.1.2.1.21.1605 =
1.3.6.1.4.1.9.9.311.1.1.2.1.22.1605 =
1.3.6.1.4.1.9.9.311.1.1.2.1.23.1605 =
1.3.6.1.4.1.9.9.311.1.1.2.1.24.1605 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.25.1605 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.26.1605 =
1.3.6.1.4.1.9.9.311.1.1.2.1.27.1605 =
1.3.6.1.4.1.9.9.311.1.1.2.1.28.1605 =
1.3.6.1.4.1.9.9.311.1.1.2.1.29.1605 = ]]]
```

SNMP Notification for a Security Event

The following example shows the SNMP notification generated for a security event (event type 411; event ID 23208). This event was generated when a new user (user1) was created on the Prime Network gateway (10.56.56.117) from the Prime Network client (10.56.20.122).

```
[V1TRAP[reqestID=0,timestamp=1 day,
14:42:23.58,enterprise=1.3.6.1.4.1.9.9.311,genericTrap=6,specificTrap=2,
VBS[1.3.6.1.4.1.9.9.311.1.1.2.1.2.1606 = 2.2
1.3.6.1.4.1.9.9.311.1.1.2.1.3.1606 = 19:26:45.52
1.3.6.1.4.1.9.9.311.1.1.2.1.4.1606 = 19:26:45.52
```

```
1.3.6.1.4.1.9.9.311.1.1.2.1.5.1606 = {[Event(Id=23208)]}
1.3.6.1.4.1.9.9.311.1.1.2.1.6.1606 = 411
1.3.6.1.4.1.9.9.311.1.1.2.1.7.1606 = 411,administrator action
1.3.6.1.4.1.9.9.311.1.1.2.1.8.1606 = 4
1.3.6.1.4.1.9.9.311.1.1.2.1.9.1606 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.10.1606 = 1,Security Event
1.3.6.1.4.1.9.9.311.1.1.2.1.11.1606 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.12.1606 = 10.56.56.117
1.3.6.1.4.1.9.9.311.1.1.2.1.13.1606 = {[BOSUser(Id=101)]}
1.3.6.1.4.1.9.9.311.1.1.2.1.14.1606 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.15.1606 = 10.56.20.122
1.3.6.1.4.1.9.9.311.1.1.2.1.16.1606 = User user1 was created
1.3.6.1.4.1.9.9.311.1.1.2.1.17.1606 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.18.1606 = 1,Information
1.3.6.1.4.1.9.9.311.1.1.2.1.19.1606 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.20.1606 =
1.3.6.1.4.1.9.9.311.1.1.2.1.21.1606 =
1.3.6.1.4.1.9.9.311.1.1.2.1.22.1606 =
1.3.6.1.4.1.9.9.311.1.1.2.1.23.1606 =
1.3.6.1.4.1.9.9.311.1.1.2.1.24.1606 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.25.1606 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.26.1606 =
1.3.6.1.4.1.9.9.311.1.1.2.1.27.1606 =
1.3.6.1.4.1.9.9.311.1.1.2.1.28.1606 =
1.3.6.1.4.1.9.9.311.1.1.2.1.29.1606 = ]]]
```

SNMP Notification for a Provisioning Event

The following example shows the SNMP notification generated for a provisioning event (event type 998; event ID 23256). This event was generated when a script (1260876995404) failed to run on the managed element (169.254.192.231) in the Prime Network gateway (10.56.56.117).

```
[V1TRAP[reqestID=0,timestamp=1 day,
16:01:53.56,enterprise=1.3.6.1.4.1.9.9.311,genericTrap=6,specificTrap=2,
VBS[1.3.6.1.4.1.9.9.311.1.1.2.1.2.1607 = 2.2
1.3.6.1.4.1.9.9.311.1.1.2.1.3.1607 = 19:34:42.20
1.3.6.1.4.1.9.9.311.1.1.2.1.4.1607 = 19:34:42.20
1.3.6.1.4.1.9.9.311.1.1.2.1.5.1607 = {[Event(Id=23256)]}
1.3.6.1.4.1.9.9.311.1.1.2.1.6.1607 = 998
1.3.6.1.4.1.9.9.311.1.1.2.1.7.1607 = 998,provisioning
1.3.6.1.4.1.9.9.311.1.1.2.1.8.1607 = 4
1.3.6.1.4.1.9.9.311.1.1.2.1.9.1607 = 7
1.3.6.1.4.1.9.9.311.1.1.2.1.10.1607 = 7,Provisioning Event
1.3.6.1.4.1.9.9.311.1.1.2.1.11.1607 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.12.1607 = 10.56.56.117
1.3.6.1.4.1.9.9.311.1.1.2.1.13.1607 =
{[ManagedElement(Key=169.254.192.231)]}
1.3.6.1.4.1.9.9.311.1.1.2.1.14.1607 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.15.1607 = 10.56.56.117
1.3.6.1.4.1.9.9.311.1.1.2.1.16.1607 = Script 1260876995404 has failed.
1.3.6.1.4.1.9.9.311.1.1.2.1.17.1607 = 5
1.3.6.1.4.1.9.9.311.1.1.2.1.18.1607 = 5,Major
1.3.6.1.4.1.9.9.311.1.1.2.1.19.1607 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.20.1607 =
1.3.6.1.4.1.9.9.311.1.1.2.1.21.1607 =
```

```
1.3.6.1.4.1.9.9.311.1.1.2.1.22.1607 =
1.3.6.1.4.1.9.9.311.1.1.2.1.23.1607 =
1.3.6.1.4.1.9.9.311.1.1.2.1.24.1607 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.25.1607 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.26.1607 =
1.3.6.1.4.1.9.9.311.1.1.2.1.27.1607 =
1.3.6.1.4.1.9.9.311.1.1.2.1.28.1607 =
1.3.6.1.4.1.9.9.311.1.1.2.1.29.1607 = ]]]
```

SNMP Notification for a System Event

The following example shows the SNMP notification generated for a system event (event type 651; event ID 23270). This event was generated when a Prime Network unit (4.5.6.7) was added to the Prime Network gateway (10.56.56.117).

```
[V1TRAP[reqestID=0,timestamp=1 day,
16:12:23.56,enterprise=1.3.6.1.4.1.9.9.311,genericTrap=6,specificTrap=2,
VBS[1.3.6.1.4.1.9.9.311.1.1.2.1.2.1612 = 2.2
1.3.6.1.4.1
.9.9.311.1.1.2.1.3.1612 = 19:35:44.75
1.3.6.1.4.1.9.9.311.1.1.2.1.4.1612 = 19:35:44.75
1.3.6.1.4.1.9.9.311.1.1.2.1.5.1612 = {[Event(Id=23270)]}
1.3.6.1.4.1.9.9.311.1.1.2.1.6.1612 = 651
1.3.6.1.4.1.9.9.311.1.1.2.1.7.1612 = 651,REGEXP:Agent
\d.*\.\d.*\.\d.*\.\d.* is starting.BOS Unit = \d.*\.\d.*\.\d.*\.\d.* AVM =
\d.*
1.3.6.1.4.1.9.9.311.1.1.2.1.8.1612 = 4
1.3.6.1.4.1.9.9.311.1.1.2.1.9.1612 = 4
1.3.6.1.4.1.9.9.311.1.1.2.1.10.1612 = 4,System Event
1.3.6.1.4.1.9.9.311.1.1.2.1.11.1612 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.12.1612 = 10.56.56.117
1.3.6.1.4.1.9.9.311.1.1.2.1.13.1612 = {[MCNetwork][MCVM(IP=4.5.6.7)]}
1.3.6.1.4.1.9.9.311.1.1.2.1.14.1612 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.15.1612 = 10.56.56.117
1.3.6.1.4.1.9.9.311.1.1.2.1.16.1612 = BOS Unit 4.5.6.7 was added and is
being started...
1.3.6.1.4.1.9.9.311.1.1.2.1.17.1612 = 4
1.3.6.1.4.1.9.9.311.1.1.2.1.18.1612 = 4,Minor
1.3.6.1.4.1.9.9.311.1.1.2.1.19.1612 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.20.1612 =
1.3.6.1.4.1.9.9.311.1.1.2.1.21.1612 =
1.3.6.1.4.1.9.9.311.1.1.2.1.22.1612 =
1.3.6.1.4.1.9.9.311.1.1.2.1.23.1612 =
1.3.6.1.4.1.9.9.311.1.1.2.1.24.1612 = 1
1.3.6.1.4.1.9.9.311.1.1.2.1.25.1612 = 0
1.3.6.1.4.1.9.9.311.1.1.2.1.26.1612 =
1.3.6.1.4.1.9.9.311.1.1.2.1.27.1612 =
1.3.6.1.4.1.9.9.311.1.1.2.1.28.1612 =
1.3.6.1.4.1.9.9.311.1.1.2.1.29.1612 = ]]]
```

SNMP Notification for Sevice Alarm with Correlation Details

The following example shows the SNMP notification generated for a service alarm (event type 17; network event ID 275049363906_1264328134638). This event was generated when the CPU utilization exceeded the upper threshold. This alarm was correlated to alarm ID 893 (see.1.3.6.1.4.1.9.9.311.1.1.2.1.29).

```
.1.3.6.1.4.1.9.9.311.1.1.2.1.2.1623 --> 2.2
.1.3.6.1.4.1.9.9.311.1.1.2.1.3.1623 --> 3 days, 0 hours, 30 minutes, 43
seconds.
.1.3.6.1.4.1.9.9.311.1.1.2.1.4.1623 --> 3 days, 0 hours, 30 minutes, 43
seconds.
.1.3.6.1.4.1.9.9.311.1.1.2.1.5.1623 -->
{[NetworkEvent(Id=275049363906_1264328134638)]}
.1.3.6.1.4.1.9.9.311.1.1.2.1.6.1623 --> 17
.1.3.6.1.4.1.9.9.311.1.1.2.1.7.1623 --> 17,cpu utilization
.1.3.6.1.4.1.9.9.311.1.1.2.1.8.1623 --> 4
.1.3.6.1.4.1.9.9.311.1.1.2.1.9.1623 --> 2
.1.3.6.1.4.1.9.9.311.1.1.2.1.10.1623 --> 2,Service Alarm
.1.3.6.1.4.1.9.9.311.1.1.2.1.11.1623 --> 1
.1.3.6.1.4.1.9.9.311.1.1.2.1.12.1623 --> 10.56.57.194
.1.3.6.1.4.1.9.9.311.1.1.2.1.13.1623 -->
{[ManagedElement(Key=169.254.192.8)]}
.1.3.6.1.4.1.9.9.311.1.1.2.1.14.1623 --> 1
.1.3.6.1.4.1.9.9.311.1.1.2.1.15.1623 --> 169.254.192.8
.1.3.6.1.4.1.9.9.311.1.1.2.1.16.1623 --> CPU utilization exceeded upper
threshold - Cleared due to ForceClear
.1.3.6.1.4.1.9.9.311.1.1.2.1.17.1623 --> 2
.1.3.6.1.4.1.9.9.311.1.1.2.1.18.1623 --> 2,Cleared
.1.3.6.1.4.1.9.9.311.1.1.2.1.19.1623 --> 0
.1.3.6.1.4.1.9.9.311.1.1.2.1.20.1623 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.21.1623 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.22.1623 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.23.1623 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.24.1623 --> 3
.1.3.6.1.4.1.9.9.311.1.1.2.1.25.1623 --> 0
.1.3.6.1.4.1.9.9.311.1.1.2.1.26.1623 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.27.1623 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.28.1623 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.29.1623 --> {[NewAlarm(Id=893)]}
```

SNMP Notification for SNMPv1 Trap with Correlation Details

The following example shows the SNMP notification generated for an SNMPv1 trap (event type 1268; network event ID 167675172732_1264334658369) for the managed element (10.56.23.124 Ethernet0/1) with the severity 5 and correlated alarm ID 916. For the trap description, see .1.3.6.1.4.1.9.9.311.1.1.2.1.16.

```
.1.3.6.1.4.1.9.9.311.1.1.2.1.2.1624 --> 2.2
.1.3.6.1.4.1.9.9.311.1.1.2.1.3.1624 --> 3 days, 2 hours, 19 minutes, 26
seconds.
.1.3.6.1.4.1.9.9.311.1.1.2.1.4.1624 --> 3 days, 2 hours, 19 minutes, 26
seconds.
.1.3.6.1.4.1.9.9.311.1.1.2.1.5.1624 -->
{[NetworkEvent(Id=167675172732_1264334658369)]}
.1.3.6.1.4.1.9.9.311.1.1.2.1.6.1624 --> 1268
.1.3.6.1.4.1.9.9.311.1.1.2.1.7.1624 --> 1268,snmp link down
.1.3.6.1.4.1.9.9.311.1.1.2.1.8.1624 --> 4
.1.3.6.1.4.1.9.9.311.1.1.2.1.9.1624 --> 5
.1.3.6.1.4.1.9.9.311.1.1.2.1.10.1624 --> 5,V1 Trap
.1.3.6.1.4.1.9.9.311.1.1.2.1.11.1624 --> 1
.1.3.6.1.4.1.9.9.311.1.1.2.1.12.1624 --> 10.56.57.194
.1.3.6.1.4.1.9.9.311.1.1.2.1.13.1624 -->
```

```
{[ManagedElement(Key=10.56.23.124)][PhysicalRoot][Chassis][Slot(SlotNum=0)
][Module][Port(PortNumber=Ethernet0/1)][Physica lLayer][Trap]}
.1.3.6.1.4.1.9.9.311.1.1.2.1.14.1624 --> 1
.1.3.6.1.4.1.9.9.311.1.1.2.1.15.1624 --> 10.56.23.124
.1.3.6.1.4.1.9.9.311.1.1.2.1.16.1624 -->
.iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifIndex.2-->2
.iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifDescr.2--
>Ethernet0/1
.iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifType.2-->6
.iso.org.dod.internet.private.enterprises.cisco.local.linterfaces.lifTable
.lifEntry.locIfR
eason.2-->administratively down

.1.3.6.1.4.1.9.9.311.1.1.2.1.17.1624 --> 5
.1.3.6.1.4.1.9.9.311.1.1.2.1.18.1624 --> 5,Major
.1.3.6.1.4.1.9.9.311.1.1.2.1.19.1624 --> 0
.1.3.6.1.4.1.9.9.311.1.1.2.1.20.1624 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.21.1624 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.22.1624 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.23.1624 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.24.1624 --> 3
.1.3.6.1.4.1.9.9.311.1.1.2.1.25.1624 --> 0
.1.3.6.1.4.1.9.9.311.1.1.2.1.26.1624 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.27.1624 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.28.1624 -->
.1.3.6.1.4.1.9.9.311.1.1.2.1.29.1624 --> {[NewAlarm(Id=916)]}
```

7 Cisco Prime Network Shell Interface

Cisco Prime Shell Interface is a deprecated interface which is not maintained in Prime Network. This interface will **not** work on a fresh installation of Prime Network. When upgrading from an old ANA version, which supported the Shell Interface, to a new Prime Network release, the Shell Interface will continue to work as before.

The information below is taken AS IS from old releases of the Integration Guide.

This section contains the following topics:

7.1 Introducing the Cisco Prime Network Shell Interface

This section describes the Cisco Prime Network shell interface, which is the command-line interface (CLI) of the Prime Network Shell Manage system. It supports a subset of the capabilities supported by the system for multilevel management of the network including VNEs, surveillance, and provisioning.

The Prime Network shell interface provides:

- A Cisco IOS-based CLI.
- A flat command hierarchy with a limited number of modes and unlimited number of nesting levels.
- Embedded inline help and command completion.
- The option to export all commands supported by the system.
- The basis for a user guide.

The Prime Network shell serves as the front-end of the system and unifies the operations of all system components for the user. The shell interface requires Prime Network system components to support the following services:

Component	Services Required from the Component
Gateway	All required surveillance and provisioning commands supported by the gateway.
AVM	Management operations at AVM level, such as starting a new
VNE	VNE management operations, such as starting and stopping VNEs.

7.1.1 Shell Users

The primary users of the Prime Network shell interface are expected to be system administrators and users who are responsible for network operation tasks, such as surveillance and provisioning.

Note All Prime Network shell commands are supported in the BQL interface. We recommend that you use BQL commands instead of Prime Network shell commands.

7.1.2 Shell Command Parameters

This section defines the different types of parameters that can be used as arguments in Prime Network shell commands. All command parameters are strings in one of the formats described in Table 7-1.

Table 7-1 Prime Network Shell Command Parameters

Parameter	Description	Format	Example
string	List of characters. If the string is not quoted, it must not contain white spaces. If the string contains white spaces, it must be quoted. To use quotes inside a string, escape them with a backslash.	—	user1 “the string \”string\””
name	The same as string. Used for clarity, for example as device name.	—	user1
integer	A string representing an integer.	—	352
vc	Identifier of an ATM VC.	<integer>[”/”<integer>]	1/102
ip (=IPAddress)	IP address.	X.X.X.X	192.168.1.2
filename	A valid filename on the operating system on which the client is running.	String	path1.snc /export/home/network310/Main/path2.snc
Date	String representing the date and time.	UNIX date format in the default C	Wed Jul 22 16:56:25 PDT 2009

7.1.3 Regular Expressions for Prime Network Shell

Prime Network shell uses the following regular expressions:

Table 7-2 Supported Regular Expressions in Prime Network Shell

Wildcard	Meaning
%	Matches any string of zero or more characters.
-	Matches any one character.

[token]	Brackets enclose ranges or sets, such as [1-9] or [klmnopq]. There are two ways to format a token:
	• Range
	Start-stop:
	  • Start is the beginning of the character range.
	  • The dash (-) is a special character indicating a range.
	  • Stop is the end of the character range.
	• Set
	Comprises discrete character values in any order. Examples:
	  • [a4Bc]
	  • [abcdefg]

[^token]	The caret (^) before a token indicates noninclusion. Example: [^c-g] means any character that is not c, d, e, f, or g.

7.2 Understanding Cisco Prime Network Shell Behavior

The Cisco Prime Network shell CLI follows guidelines similar to the Cisco IOS interface. The interface includes the following:

- Command completion—Pressing the Tab key completes a partially typed command. If more than one valid completion exists, the Prime Network shell beeps.
- Completion sound notification—Short alarm notice is given when completion is not available.
- Fast help mechanism—Entering ? lists all valid completions.
- Error messages—See Cisco Prime Network Shell Errors, page 24-1.
- Case insensitive interface.
- Partial syntax recognition.
- Validation of the input.
- Use of Prime Network shell syntax when connecting to Prime Network Administration and unit machines—Using Telnet to connect to a unit allows all management functions relevant to a single machine.
- Support for the More command and terminal length.

Note All Prime Network shell commands are supported in the BQL interface. We recommend that you use BQL commands instead of Prime Network shell commands.

The following section includes:

7.2.1 Prime Network Shell Prompt and Nodes

After logging into Prime Network gateway enter the following command to get into the Prime Network shell mode (entrance mode):

```
# su - bos
        Sheer Networks
BOS Starting up...
MetroShell>
```

Table 7-3 identifies and describes the four Prime Network shell modes:

Table 7-3 Prime Network Shell Modes

Mode	Command	Prompt	Description
exec	—	MetroShell>	Entrance mode—Designed to show general details to the operator user.
enable	enable	MetroShell#	Enhanced user mode—Allows further details and configuration. Activation of this mode is authenticated using a password.
configure	configure or config	MetroShell(CONFIG)#	Configuration mode—Used to set different attributes. Activation of this mode is authenticated in and possible only from enable mode. You must be in the enable mode before changing to this mode.

manage	manage	MetroShell(MANAGE)#	Management mode—Used to perform management operations on the system. This mode requires authentication. You must be in the enable mode before changing to this mode.

Note Each level includes all previous levels. Enter ? to review all supported commands for each of the modes.

7.2.2 Output Format

This section describes the format of the output that the Prime Network shell can return and includes the following topics:

- Table, page 401
- Paragraph, page 401

Some commands provide no output, such as the exit command.

Table

Prime Network shell can produce output in a table with the following attributes:

- A header that:
  - Lists the names of the columns.
  - Appears only once at the beginning of the table, and not on each page.
- A horizontal ruler that separates the header from the rows of the table.
- Left-aligned text in all table fields. For example:

No.	IP	Name	Type	Uptime
1	192.168.2.3	asam1	ASAM1000	29.04.02 13:12
10	192.168.2.4	asam2	ASAM1000	1.05.02 9:43
11	192.168.2.45	RedBack2	SMS500	1.05.02 9:44
100	192.168.2.46	RedBack3	SMS500	1.05.02 9:44

Paragraph

Prime Network shell can produce output in text paragraphs that use the following format:

```
<objectname>:
  <attribute name> = <value>
  ...
```

For example:

```
192.168.2.3:
  IP address = 192.168.2.3
  Name = asam1
  Type = ASAM1000
  Vendor = Alcatel
  Uptime = 1.05.02 13:13
Status = OK
```

7.2.3 Output Redirection

Output redirection enables you to send the output of a command to a file. Prime Network shell provides two options for output redirection:

- > filename—Enter at the end of the command to create a new file and redirect the command output to it. If the file already exists, the new file overwrites the old one. If an error occurs when creating the output file, the command is not run.
  Example: show device > out.txt
- >> filename—Enter at the end of the command to append the output to an existing file. The file is created if it does not already exist.
  Example: show device >> devices0515.text

7.2.4 Background Processing

Each Prime Network shell command may be run in the background by including an ampersand (&) at the end of the command. For example, entering the command show link > links.txt & runs the show link command in the background and redirects all output to the file links.txt. By default, unless redirection is specified, the output of a background command is redirected to the Prime Network shell terminal.

7.2.4.1 Basic Commands

This section describes the following basic commands:

Inline help

Property	Description
Name	Inline help.
Description	Supplies command completion while typing.
Mode	All modes.
Usage	`?`
General	The command executes without pressing Enter.
Example	`show ?`
Output Format	Paragraph: • Lists the valid options with a short description for each option. • If the description exceeds a single line, the lines after the first one are indented with the first description line.

Exit Prime Network Shell

Property	Description
Name	Exit Prime Network shell.
Description	Exit Prime Network shell interface.
Mode	`exec`
Usage	`exit`
General	If the Prime Network shell serves as the shell for the machine, it returns to the login window.
Example	`exit`
Output Format	None.

Exit Current Mode

Property	Description
Name	Exit current mode.
Description	Exit current Prime Network shell mode and return to the previous mode.
Mode	`enable, config, manage`
Usage	`exit`

Property	Description
General	Running exit in enable mode returns to exec mode. Running exit in config mode returns to enable mode. Running exit in manage mode returns to the mode from which the user entered it, either enable or configure.
Example	`exit`
Output Format	None.

Help

Property	Description
Name	Help.
Description	Prints general help. Lists all the commands with a short description for each command.
Mode	All modes.
Usage	`help`
General	—
Example	`help`
Output Format	Paragraph containing a fixed help message.

Terminal Length

Property	Description
Name	Terminal length.
Description	Set terminal length.
Mode	`all modes`
Usage	`terminal length <integer>`
General	Use length 0 for no pausing.
Example	`terminal length 40`
Output Format	None.

Show History

Property	Description	
Name	Show history.	
Description	Show previously run command.	
Mode	`all modes`	
Usage	`history`	
General	The history contains the last 100 commands. This is not configurable.	
Example	`history`	
Output format	Table	
	Column	**Description**
	Index	The index of the command. Index 1 refers to the previous command.
	Command	The string of the command.

Clear History

Property	Description
Name	Clear history.
Description	Clear the command list stored in the history buffer.
Mode	`all modes`
Usage	`history clear`
General	—
Example	`history clear`
Output Format	None.

Access History

Property	Description
Name	Access history.
Description	Runs against a command in the history buffer.
Mode	`all modes`

Property	Description
Usage	`history <integer>`
General	Specifying 1 indicates the last command (not including the current history command), specifying 2 indicates the command before that, and so on.
Example	`history 3`
Output Format	None.

Execute Script

Property	Description
Name	Execute script.
Description	Run a script file of Prime Network shell commands.
Mode	All modes.
Usage	`run <filename> [async] [silent]`
General	The scripts must reside on the UNIX machine running the Prime Network shell in the directory /Main. This directory resides under the directory where the system was installed. The files should be transferred to this directory or its subdirectories using FTP. If a script resides in a subdirectory of /Main, include the relative path with the name of the script. The silent option suppresses any output that the command sends to the terminal. **Note** You can run the script in the background by appending & to the command.
Example	`run provision.cmd` `run scripts/provision.cmd`
Output Format	None.

7.2.5 Unit Management

This section describes the commands needed to manage a unit. The Unit management includes management of the AVM processes and VNEs within the AVMs. Topics include:

- AVM Management, page 407
- VNE Management, page 408

7.2.5.1 AVM Management

AVM management commands include:

- Show AVM List, page 407
- Show AVM VNEs, page 407

Show AVM List

Property	Description	
Name	Show AVM list.	
Description	Show a list of AVMs with their minimum set of properties.	
Mode	`manage`	
Usage	`show [unit [<IPAddress>]] avm`	
General	If an IP address is specified, this command shows AVMs on the indicated machine only. If no IP address is specified, this command shows AVMs on all machines. If no unit is specified, the command refers to the current machine.	
Example	`show unit avm`	
Output Format	Table	
	Column	**Description**
	Machine	IP address of the machine where the AVM resides.
	ID	AVM ID.
	PID	Process ID.
	Port	Management port.
	Uptime	Process uptime (date format).
	Version	AVM version.

Show AVM VNEs

Property	Description
Name	Show AVM VNEs.
Description	List all VNEs for a specific AVM.
Mode	`manage`

<table>
<tr><th>Property</th><th>Description</th></tr>
<tr><td>Usage</td><td><pre>show [unit <IPAddress>] avm <integer> | all agent [detailed]</pre></td></tr>
<tr><td>General</td><td>Lists all VNEs for the specified AVM. If no unit is given, the command refers to the current machine.

If detailed is not specified, only DAs are displayed. Otherwise, all agent types (Device Agent [DA], Collector Agent [CA], and Instrumentor Agent [IA]) are displayed.

The argument all refers to all AVMs in the current machine.

The command also displays configured VNEs, which are configured in the XML but are not loaded. In this scenario, all nonrelevant fields are empty.</td></tr>
<tr><td>Example</td><td><pre>show unit 192.168.2.10 avm 32 agent</pre></td></tr>
<tr><td>Output format</td><td>Table
<table>
<tr><th>Column</th><th>Description</th></tr>
<tr><td>IP address</td><td>VNE leading IP address.</td></tr>
<tr><td>Type</td><td>Agent type: DA, IA, or CA.</td></tr>
<tr><td>State</td><td>VNE state: idle, wait, block, running, or configured.</td></tr>
<tr><td>Runtime</td><td>The total time spent by the VNE processing messages.</td></tr>
<tr><td>Wait time</td><td>The total time spent by the VNE waiting to process messages.</td></tr>
<tr><td>Last run</td><td>The last time the VNE visited the scheduler. Units are milliseconds relative to now.</td></tr>
<tr><td>Transport address</td><td>The VNE transport address in hexadecimal format.</td></tr>
<tr><td>Parent</td><td>Parent VNE. Transport address of the parent VNE.</td></tr>
</table></td></tr>
</table>

7.2.5.2 VNE Management

VNE management commands include:

- Show All VNEs, page 409
- Show VNE Information, page 409
- Add AVM, page 410
- Remove AVM, page 410

Show All VNEs

Property	Description
Name	Show all VNEs in the unit.
Description	Show the basic information for all VNEs for all AVMs on the unit.
Mode	`manage`
Usage	`show agent [detailed]`
General	If detailed is not specified, only DAs are displayed. Otherwise, all agent types (DA, CA, and IA) are displayed.
Example	`show agent`
Output Format	See Show AVM VNEs, page 407.

Show VNE Information

Property	Description	
Name	Show VNE information.	
Description	Show the information for a specific VNE.	
Mode	`manage`	
Usage	`show agent <IPAddress | name>`	
General	The parameter can be the leading IP address of the VNE or the device name.	
Example	`show agent 192.168.2.2`	
Output Format	Paragraph.	
	Field	**Description**
	IP Address	VNE leading IP address.
	Type	Agent type: DA, IA, or CA.
	Machine	IP address of the machine where the VNE is installed.

Property	Description	
	AVM	AVM number where the VNE is installed.
	Transport address	The transport address of the VNE in hexadecimal format.
	State	VNE state: idle, wait, block, running, or configured.
	Runtime	The total time spent by the VNE processing messages.
	Wait time	The total time spent by the VNE waiting to process messages.
	Last run	The last time the VNE visited the scheduler. Units are milliseconds relative to now.
	Parent	Parent VNE. Transport address of the parent VNE.

Add AVM

Property	Description
Name	Add AVM.
Description	Add a new AVM to a unit.
Mode	`manage`
Usage	`unit <IPAddress> avm <integer> add`
General	The integer parameter represents the ID of the AVM to be added.
Example	`unit 192.168.2.10 avm 32 add`
Output Format	None.

Remove AVM

Property	Description
Name	Remove AVM.
Description	Remove an AVM from a unit machine.
Mode	`manage`
Usage	`unit <IPAddress> avm <integer> remove`
General	The integer parameter represents the ID of the AVM to be deleted.

Property	Description
Example	`unit 192.168.2.10 avm 32 remove`
Output Format	None.

Load AVM

Property	Description
Name	Load AVM.
Description	Add a configured AVM to the unit bootstrap list.
Mode	`manage`
Usage	`unit <IPAddress> avm <integer> load`
General	The integer parameter represents the ID of the AVM to be loaded. The newly loaded AVM starts immediately and will be started in all consecutive restarts of the system.
Example	`unit 192.168.2.10 avm 32 load`
Output Format	None.

Unload AVM

Property	Description
Name	Unload AVM.
Description	Remove an AVM from the bootstrap list.
Mode	`manage`
Usage	`unit <IPAddress> avm <integer> unload`
General	The AVM is automatically stopped if it is currently executing. The integer parameter represents the ID of the AVM to be unloaded.
Example	`unit 192.168.2.10 avm 32 unload`
Output Format	None.

Add VNE

Property	Description
Name	Add VNE.
Description	Add a VNE to the configuration database of a given AVM.
Mode	`manage`

Property	Description
Usage	`agent <IPAddress> add unit <IPAddress> avm <integer> name <name> vendor <string> type <string>`
General	If no name is given, the IP address is used as the device name.
Example	`agent 192.168.2.3 add unit 192.168.2.10 avm 32 vendor alcatel type asam1000`
Output Format	None.

Remove VNE

Property	Description
Name	Remove VNE.
Description	Remove a VNE from a given AVM.
Mode	`manage`
Usage	`agent <IPAddress> remove`
General	If the VNE is currently running, it stops. If the VNE is loaded, it is unloaded from the bootstrap list.
Example	`agent 192.168.2.3 remove`
Output Format	None.

Load VNE

Property	Description
Name	Load VNE.
Description	The newly loaded VNE starts immediately and will be loaded every time the system restarts.
Mode	`manage`
Usage	`agent <IPAddress> load`
General	—
Example	`agent 192.168.2.3 load`
Output Format	None.

Unload VNE

Property	Description
Name	Unload VNE.

Property	Description
Description	Unload an VNE from the AVM bootstrap list. If the VNE is currently running, it is stopped before it is unloaded from the bootstrap list.
Mode	manage
Usage	agent <IPAddress> unload
General	—
Example	agent 192.168.2.3 unload
Output Format	None.

Add Static Topology Link

Property	Description
Name	Add static topology link.
Description	Add a static link between two devices in the network.
Mode	manage
Usage	topology link source <IPAddress> [shelf <integer>] module <integer> [submodule <integer>] port <integer> destination <IPAddress> [shelf <integer>] module <integer> [submodule <integer>] port <integer> add [unidirectional]
General	By default, the link is bidirectional and enabled automatically. Unidirectional represents a unidirectional link.
Example	topology link source 192.168.2.3 module 1 port 1 destination 192.168.2.4 module 2 port 1 add
Output Format	None.

Remove Static Topology Link

Property	Description
Name	Remove static topology link.
Description	Remove an existing static topology link.
Mode	manage
Usage	topology link source <IPAddress> [shelf <integer>] module <integer> [submodule <integer>] port <integer> destination <IPAddress> [shelf

Property	Description
	`<integer>] module <integer> [submodule <integer>] port <integer> remove [unidirectional]`
General	—
Example	`topology link source 192.168.2.3 module 1 port 1 destination 192.168.2.4 module 2 port 1 remove`
Output Format	None.

7.2.6 Surveillance

This section describes the surveillance commands that are supported by the Prime Network shell interface. The surveillance commands include:

- Show Links, page 414
- Drools Rules Management, page 414

7.2.6.1 Show Links

Property	Description	
Name	Show links.	
Description	Show the topological links managed by the unit.	
Mode	`enable`	
Usage	`show link`	
General	—	
Example	`show link`	
Output Format	Table.	
	Column	**Description**
	Index	Unique running index.
	From	A-side location.
	To	Z-side location.
	State	Automatic, Static, Configured.

7.2.6.2 Drools Rules Management

For more information about the Drools Rules engine, see Cisco Prime Network 4.2.2 Customization User Guide.

Topics in this section include:

- Show All Rules, page 415

Show All Rules

Property	Description
Name	Show rules.
Description	Show all rules.
Mode	`enable`
Usage	`show rule`
General	—
Example	`show rule`
Output Format	contextID, ruleName, isValid.

Show Specific Rules

Property	Description
Name	Show rules.
Description	Show rules of a specific context.
Mode	`enable`
Usage	`show rule <contextID>`
General	—
Example	`show rule pre`
Output Format	ContextId, Rule Name, is Valid.

Reload Rules

Property	Description
Name	Reload rules.
Description	Reloads all rules of a specific context.
Mode	`config`
Usage	`rule <contextID> reload`
General	—

Property	Description
Example	`rule Post reload`
Output Format	—

Validate Rule

Property	Description
Name	Validate rule.
Description	Validate a specific rule.
Mode	`config`
Usage	`rule <contextID> <ruleName> validate`
General	—
Example	`rule pre sendGenericEvents validate`
Output Format	—

7.3 Cisco Prime Network Shell Errors

This section includes the following topics concerning shell errors:

Error Format

Upon function termination with an error, the error uses the following format: ERROR (error code): error message
For example:

```
ERROR(10443): IP address already in use
```

If a parsing or type-check error occurs (for example, entering a string instead of an integer), the command is reprinted with an arrow pointing to the erroneous phrase. If it is not a parsing or type-check error, the command is not reprinted.
For example, if you enter show ip 192.168.1, the result is:

```
ERROR (203): Invalid Value show ip 192.168.1
```

Command Completion Errors

Error Example	Error Message

show momomo ^	Unknown command—A caret (^) marks the first letter that is unrecognized.
ip change <cr>	Incomplete command—When more arguments are needed.
s <TAB>	Beep if more than one command starts with an s.

Error codes

Code	Error Constant	Description
0	NO_ERROR	Operation completed successfully.
1000	GENERAL_ERROR	General error. This is the most generic error and should be reported only when a more concrete error code does not exist.
2000	EXECUTION_FAILED	General error caused by an error while trying to execute a command. This can occur, for example, because the VNE does not exist or because an incorrect parameter was specified.
3000	CONNECTION_FAILD	General connection failure. Use more concrete subtypes when possible.
3100	CONNECTION_WITH_MM_FAILED	Prime Network could not connect to the Prime Network gateway.
3101	CONNECTION_WITH_MC_FAILED	Prime Network could not connect to a Prime Network unit.
4000	COMMAND_NOT_SUPPORTED	The command is not supported by the MM or the unit.
5000	INVALID_VALUE	General error for an invalid parameter value.

8 Appendixes

This section contains the following topics:

- Appendix A, “Cisco Prime Network GUI and BQL Mapping”
- Media Types to Poll Inventory Information

This section describes media type integer mapping details to poll the inventory information from Prime through NBI.

Media Type	Integer Mapping Value	Description
COAX	1	Coax
Thin_COAX	2	Thin coax
Thick_COAX	3	Thick coax
Fiber_OPTIC	4	Fiber Optic
MULTIMODE_FO	5	Multi mode fiber optic
SINGLEMODE_FO	6	Single mode fiber optic
SHORT_SINGLEMODE_FO	7	Short single mode fiber optic
LONG_SINGLEMODE_FO	8	Long single mode fiber optic
UTP	9	UTP
STP	10	STP
FTP	11	FTP
EIA_TIA_232	12	
EIA_TIA_449	13	
V_35	14	
X_21	15	
EIA_TIA_530	16	
EIA_TIA_530A	17	
GENERIC_SERIAL	18	
EIA_TIA_612_613	19	

- Appendix B, “Productivity Tools”
- Appendix C, “Change the Root-Cause Analysis Mechanism”

8.1 Appendix A, “Cisco Prime Network GUI and BQL Mapping”

The following tables maps the BQL commands with the Cisco Prime Network GUI operations:

- Table A-1: Cisco Prime Network Vision GUI to BQL Mapping
- Table A-2: Cisco Prime Network Events GUI to BQL Mapping
- Table A-3: Cisco Prime Network Administration GUI to BQL Mapping

Table A 1 maps the BQL commands with the Prime Network Vision GUI operations.

Table A 1 Cisco Prime Network Vision GUI to BQL Mapping

GUI Operation	Corresponding APIs		
	Command	Parameter Type	Value
Maps			
Load maps	Find[1]	Imo	IMap
Open map	Get	Oid	map-oid
Create a new map	Create	Imo	IMap
Add device	CreateAndAdd	parentOid	Node-oid
Remove device	Delete	Oids	Node-oid
Save a map	SaveMapAspect	dummyOid	{[]}
		imobjectArr	<IMapAspect type="**IMapAspect**" instance_id="0"> <ID type="Oid">{[Hierarchy Node(Id=1038)][MapAspect]}</ID> . . . </IMapAspect>
Rename a map	Update	Oid	map-oid
Delete a Map	Delete	Oid	map-oid
Update map options	Update	Oid	mapdataaspect-oid
Device Properties			
View VNE properties	GetElement	Oid	me-oid
Network Device			
View physical and	Get	Oid	me-oid
Enable or disable sending alarms	EnableSending Alarms	Oid	port-oid
Links			
View link properties	Get	Oid	topology-link-oid
Tickets			
• View ticket properties • View affected parties	Get	Oid	ticket-oid

GUI Operation	Corresponding APIs		
	Command	Parameter Type	Value
Acknowledge a ticket	Acknowledge	Oid	ticket-oid
Clear a ticket	ForceClear	Oid	ticket-oid
Remove a ticket	Remove	Oid	ticket-oid
Path Tracer			
Path trace to IP destination	GetSNC	startingPoint, layer3Data	data-link-layer-oid destination ipaddress
Business Tags			
Attach business tag	Create	imobject	ibusinessobject-imo-attributes
Edit business tag	Update	Oid	businessobject-oid
Detach business tag	Delete	Oid	businessobject-oid
Find business tag	Find1	Imo	ibusinessobject-imo-attributes
Command Builder			
• New command • Edit command • Import command	Set	imo,neoid	iscript-imo-attribute, me-oid
Preview execution	PreviewScript	cid	command-oid
Execute command	command-name	oid,	me-oid,
Delete command	Delete	Oid	script-oid
Soft Properties			
View all soft properties	GetAllSoft PropertiesForIMO	Ne	me-imo-attributes
• New soft property • Edit soft property • Import property	Set	Imo	softproperty-imo-attributes
Debug soft property	DebugSoftProperty	Imo	softproperty-imo-attributes
Stop debug soft property	StopDebugSoftProperty	Imo	softproperty-imo-attributes
Delete soft property	Delete	Oid	softproperty-oid
Report Manager			

GUI Operation	Corresponding APIs Command	Parameter Type	Value
Add a new report folder	AddReportCategory	—	{[ReportRoot]} or {[ReportRoot][ReportCategory(Category=*ReportCategory*)]}
Create a new user-defined report	SaveReportTypeCommand	—	{[ReportRoot][ReportCategory(Category=*ReportCategory*)]}
Generate a report	RunReportTypeCommand	—	—
Rename a report	Rename	—	[ReportCategory(Category=*ReportCategory*)][ReportCategory(Category=*User-definedReportName*)]
Move a user-defined report	Move	—	[ReportRoot][ReportCategory(Category=*ReportCategory*)][ReportData(Id= *ReportID*)]
Delete a report	Delete	—	[ReportCategory(Category=*ReportCategory*)][ReportType(Type=*ReportType*))]})][Report(Id=*Report ID*)]
Delete a report folder	Delete	—	[ReportCategory(Category=*ReportCategory*)][ReportCategory(Category=*User-defined Reports*)]

1. Retrieves information for objects according to search criteria.

Table A 2 maps the BQL commands with the Prime Network Events GUI operations.

Table A 2 Cisco Prime Network Events GUI to BQL Mapping

GUI Operation	Corresponding APIs		
	Command	Parameter Type	Value
Prime Network Tickets and Events Properties			
View events and tickets	Get	Oid	{[TicketListAspect]}
	Get	Oid	{[NewAlarm(id=<AlarmID>]}
	Get	Oid	{[Event(Id=<EventId>)]}
	Get	Oid	{[NetworkEvent(Id=<Network EventID>)]}

Table A 3 maps the BQL commands with the Prime Network Administration GUI operations.

Table A 3 Cisco Prime Network Administration GUI to BQL Mapping

GUI Operation	Corresponding APIs		
	Command	Parameter Type	Value
Deploy Prime Network			
View network	Get	Oid	{[MCNetwork]}
Managing Units			
Add units	Create	Oid	unit-oid
Edit unit properties	Update	Oid	unit-oid
Delete units	Delete	Oid	unit-oid
View units	Get	Oid	unit-oid
Managing AVMs and VNEs			
Create AVMs	Create	Oid	avm-oid
View AVM properties	Get	Oid	avm-oid
Edit AVM properties	Update	Oid	avm-oid
Delete an AVM	Delete	Oid	avm-oid
Start AVMs	Load	Oid	avm-oid
Stop AVMs	Unload	Oid	avm-oid
Move AVMs	Update	Oid	avm-oid
Find AVMs	Get	Oid	avm-oid
Create VNEs	Create	imobject	element-mgmt-imo
View VNE properties	Get	Oid	element-mgmt-oid
Edit VNE properties	Update	Oid	element-mgmt-oid
Delete a VNE	Delete	Oid	element-mgmt-oid

GUI Operation	Corresponding APIs		
	Command	Parameter Type	Value
Change the VNE state	Update	Oid	element-mgmt-oid
Move multiple and single VNEs	Update	Oid	element-mgmt-oid
Managing Global Settings			
View client license properties	Get	Oid	{[ClientLicensesManagement]}
View database segments	Get	Oid	{[DBSegmentRoot]}
Get message of the day	Get	Oid	{[MessageOfTheDay]}
Customize a message of the day	Update	Oid	{[MessageOfTheDay]}
View polling groups	Get	Oid	{[PollingGroupsManagement
New polling group	Create	imobject	pollinggrpmgmt-imo-attributes
Modify polling group	Update	Oid	pollinggrpmgmt-oid
Modify polling interval	Update	Oid	{[PollingInterval]}
Delete polling group	Delete	Oid	pollinggrpmgmt-oid
Create a protection group	Update	Oid	{[ProtectionGroups]}
View protection group properties	Get	Oid	{[ProtectionGroups]}
Edit protection group properties	Update	Oid	protectiongrp-oid
Delete a protection group	Update	Oid	{[ProtectionGroups]}
Managing Links			
View static links	Get	Oid	{[StaticTopologyManagement]}
Create a static link	Update	Oid	{[StaticTopologyManagement]}
Delete a static link	Update	Oid	{[StaticTopologyManagement]}
Managing Workflows			
Template management	Get	Oid	{[WorkflowTemplateManagement]}

GUI Operation	Corresponding APIs		
	Command	Parameter Type	Value
Workflow management	Get	Oid	{[WorkflowManageme nt]}
Managing Security			
Create scope	Create	imobject	scope-imo-attributes
View scope properties	Get	Oid	{[ScopeRoot]}
Edit scope properties	Update	Oid	scope-oid
Delete scope	Delete	Oid	scope-oid
Create user	Create	Oid	bosuser-oid
View user	Get	Oid	{[UserRoot]}
	Get	Oid	bosuser-oid
Grant or edit user	Update	Oid	bosuser-oid
Change user password	Update	Oid	bosuser-oid
Delete user	Delete	Oid	bosuser-oid
Event Notification Service			
Create a subscription to receive notifications.	Create	Oid	[OssClientInfoRoot] [OSSClientInfo]
Update the subscription	Update	Oid	[OssClientInfoRoot] [OSSClientInfo(Id=*Registration ID Number*)]}
Remove the subscription based on the specified OID.	Delete	Oid	[OssClientInfoRoot] [OSSClientInfo(Id=*Registration ID Number*)]

8.2 Media Types to Poll Inventory Information

This section describes media type integer mapping details to poll the inventory information from Prime through NBI.

Media Type	Integer Mapping Value	Description
COAX	1	Coax
Thin_COAX	2	Thin coax
Thick_COAX	3	Thick coax
Fiber_OPTIC	4	Fiber Optic
MULTIMODE_FO	5	Multi mode fiber optic
SINGLEMODE_FO	6	Single mode fiber optic
SHORT_SINGLEMODE_FO	7	Short single mode fiber optic

LONG_SINGLEMODE_FO	8	Long single mode fiber optic
UTP	9	UTP
STP	10	STP
FTP	11	FTP
EIA_TIA_232	12	
EIA_TIA_449	13	
V_35	14	
X_21	15	
EIA_TIA_530	16	
EIA_TIA_530A	17	
GENERIC_SERIAL	18	
EIA_TIA_612_613	19	

8.3 Appendix B, "Productivity Tools"

This section describes the tools that are packaged as part of Cisco Prime Network. These tools can be used to create BQL commands and identify OID types and values. You can use the productivity tools to perform the following:

- Redirect the BQL output to an output file.
- Generate the Prime Network inventory report and export it to an excel file.

Note the following:

- The debugging tools are intended for support purposes only. It is recommended that you use these tools judiciously as they can have a negative effect on performance and scale.
- Loading the debugging tools causes them to store and process additional metadata. Hence, it is recommended that you do debugging on a separate GUI instance; or at least disable and clear all the information in the debug views before dismissing the view, because even if the view is closed, the debugging tools continue to detect internal changes.

8.3.1 Mediator Debugger

Mediator Debugger is a tool that helps you identify the BQL commands for any Prime Network GUI operation. Using this tool, you can write your own BQL commands for the required GUI operations. To run Mediator Debugger:

1. Log into either of the Prime Network client GUI applications (Cisco Prime Network Vision or Cisco Prime Network Administration).
2. Press Ctrl + F11. The Mediator Debugger window appears. Click Enabled and Follow Additions.

3. Perform any tasks using the Prime Network client. You can view the respective BQL command details in the Mediator Debugger window under the following tabs:
   - CID—Displays the command invoked for that particular GUI operation.
   - Result—Displays result of the command execution.
   - Time—Displays the start and end time of the command execution.
   - Events—Displays events received for the given command. Events can be either the result of a command or a notification if you have registered for notification service When you select an event, the Debug tab shows the contents of the event as a Java Object dump, and the BQL tab shows the contents of the event in XML format.
   - Current—Displays the command result with IMO details.

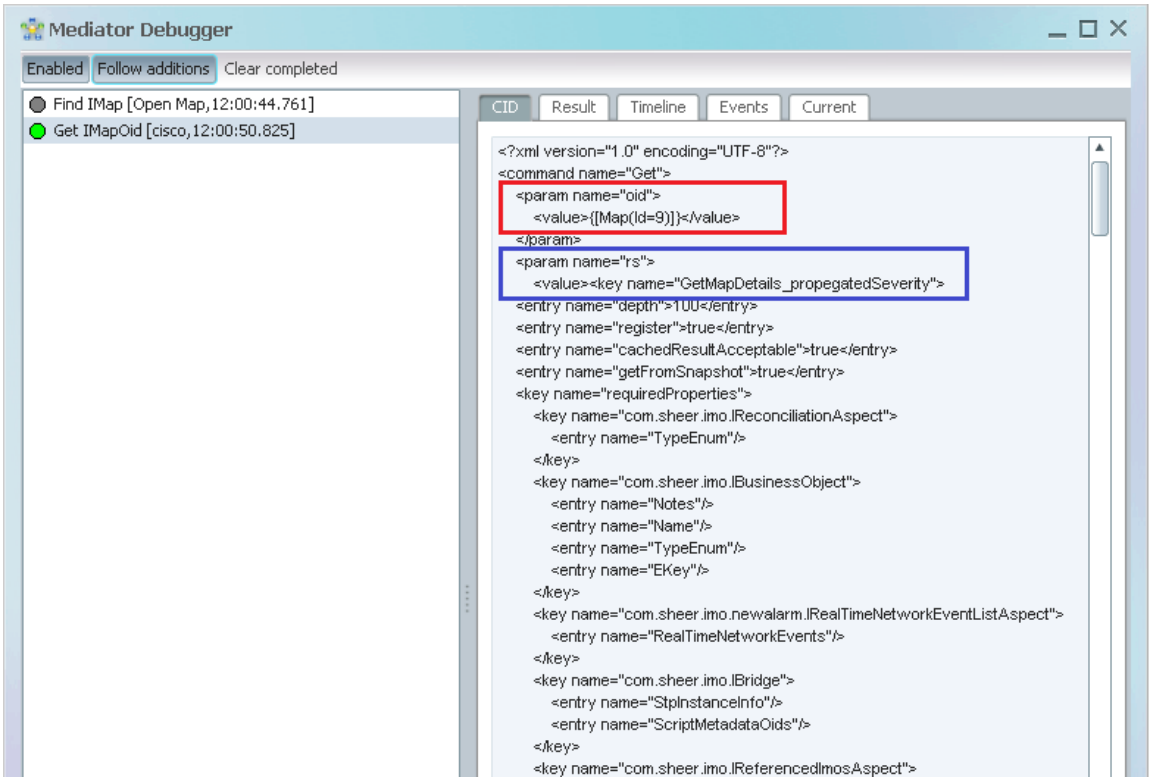
After copying the command to a text editor, you can modify and run the command to perform the same operation using BQL. See BQL Command Format, page 5-1 for details.

8.3.1.1 Determining OID and rskey

To determine the OID and rskey to view a specific map:

1. Log into the Prime Network Vision.
2. Press Ctrl + F11. The Mediator Debugger window appears.
3. Click Enabled and Follow Additions.
4. Click File > Open. The Open Map window is displayed. Choose a map and click OK. The selected map appears in the Prime Network Vision window.
5. Go to the Mediator Debugger window and select Get IMapOid.

Figure B 1 Mediator Debugger Window with OID and rs key Information



6. View the details displayed for the param name=”oid” and param name=”rs” parameters in the CID pane.
   The value for these parameters determines the OID and rs key. In this example, the OID is [Map(Id=9)] and the rskey is GetMapDetails. See Figure B-1.

   Prime Network 4.0 supports multiple contexts for VNEs. As a result, the object identifier (OID) of elements that can be assigned to a context now includes the context name. This change includes most of the logical inventory OIDs in this release.

For example, the bridge OID old format:

```
{[ManagedElement(Key=vne_name)][LogicalRoot][FWComponentContainer(Type=
2)][Bridge(BridgeNa me=bridge_name)]}
```

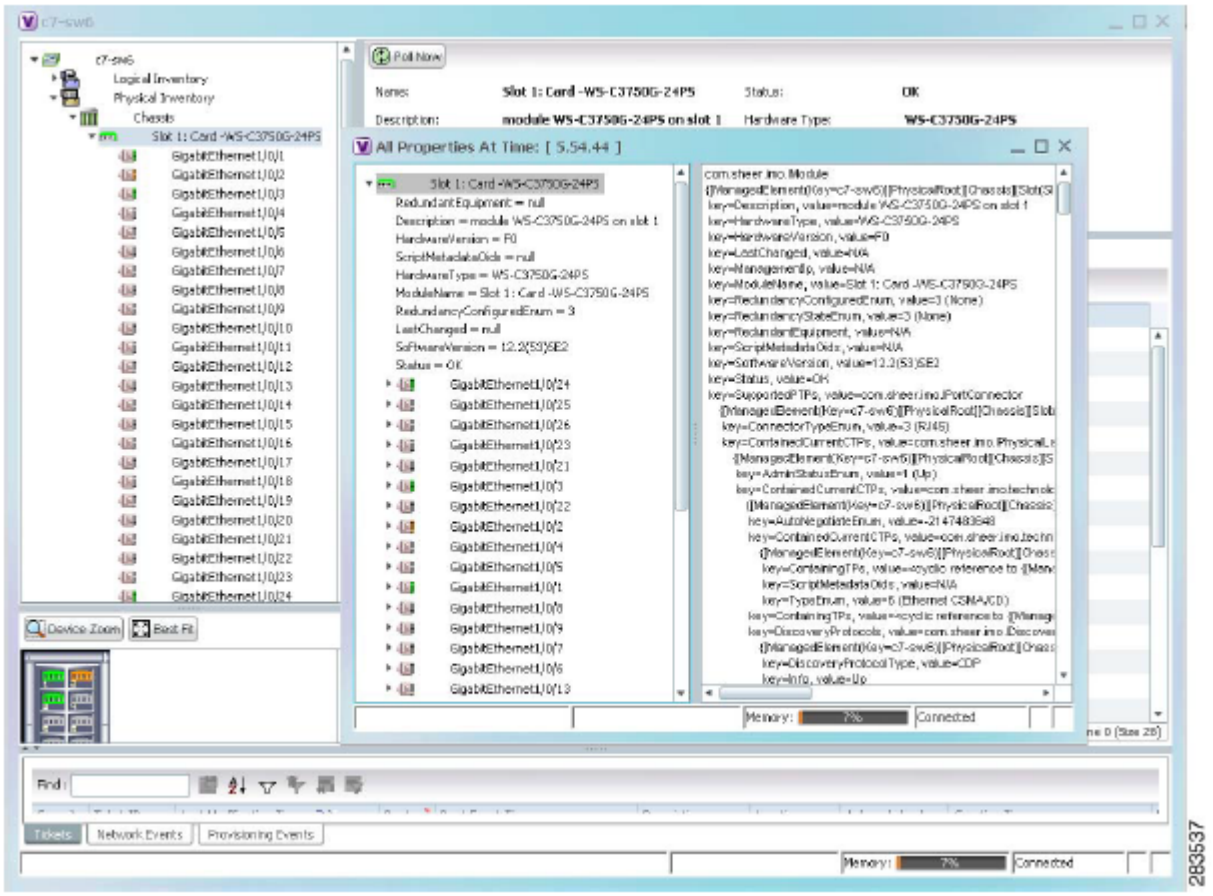
In now represented as:

```
{[ManagedElement(Key=
vne_name)][LogicalRoot][Context(ContextName=Default
context)][FWComponentContainer(Type=2)][Bridge(BridgeName=bridge_name)]
}
```

8.3.2 Viewing IMO in the GUI

You can view the IMO and its attributes by using the F2 key on the Prime Network client applications. The All Properties At Time window displays the OID, object types, and object containment hierarchy.

Figure B 2 IMO Browser - All Properties At Time Window



8.3.3 Redirecting BQL result to output file

You can use a Perl script that reads a BQL command from an input file and produces a file with the BQL result. This Perl script connects to a Prime Network gateway and runs a BQL command from a text file. The BQL result is then written to an output file.
You can find this script file under ~/Main/scripts.

To install and run the script:

1. Move the script file to a directory on the relevant Prime Network gateway.
2. Grant executable privileges to the script by using the following command:
   `(chmod +x runBQL.pl)`
3. Edit the runBQL.pl script to change the server credentials such as username and password.
4. Edit the BQLCommand.txt file to contain the BQL command, which you want to run.
5. Run the following command:
   `perl ./runBQL.pl BQLCommand.txt > output file`

   where BQLCommand.txt is the text file that contains the BQL command, which you want to run; and output file is the file where you want to save the BQL result.

8.3.4 Generating the Prime Network Inventory Report

You can use a Perl script to query the Prime Network inventory and export the result to a combined XML file. This Perl script loops across all or selected VNEs, and executes a given command into an output XML file. You can find this script on the Cisco Developer Network in the following path:
https://developer.cisco.com/site/prime-network/

To install and run the script:

1. Extract the .zip file and save the Perl script on a client, where Perl is installed. This step is not required is you choose to install it on the server.
2. Edit the Bql-network-inv.pl script to change the server IP address and credentials such as username and password.
3. Create a BQL input file with the BQL command, which you want to run.
4. (Optional) Generate an input file to filter the query, by using one of the following options:
   a. You can use the extracted optional script Bql-create-dev-list.pl to retrieve a device list from Prime Network in regular XML format. To do so:
      - Edit the Bql-create-dev.pl script to change the server IP address and credentials such as username and password.
      - Run the following script:
        `perl ./Bql-create-dev-list.pl output-file`
        where, output-file is the name of the file where you want to save the BQL result.
      - This file can be filtered later to contain only a partial set of devices.

b. You can generate an input file in any other format as long as the device keys are specified in OID tokens ({[ManagedElement(Key=XXXXX)]}) since the file is parsed via regular expressions. Run the following command:

```
perl ./Bql-network-inv.pl bql-file input file
```

where,

- bql-file is a mandatory text file with the BQL command. The start point OID of this command must be IManagedElement or any node below it in the physical or logical inventory tree hierarchy.
- input file is an optional parameter that allows you to limit the query to a selected set of VNEs. By default (when such a file is not provided), the command queries the entire device list from Prime Network and uses it for generating the report. The output file (Network-inventory.xml) is placed under the same directory as the script(s).

8.3.5 Exporting the Prime Network Inventory Report to an Excel File

You can use a Perl script to generate a Microsoft Office Excel report that lists the physical inventory in Prime Network. You can view the physical inventory properties for all devices down to the module level.

Note Ports will not be included in the report.

Tip Office Excel has an automatic XML parsing mechanism. Using this mechanism, you can import any BQL report output into an Office Excel file and it will automatically be parsed and flattened into a table.

You can find this script on the Cisco Developer Network in the following path:

https://developer.cisco.com/site/prime-network/tips-and-tools/productivity.gsp

To install and run the script:

1. Extract the .zip file and save the Perl script on a client, where Perl is installed. This step is not required is you choose to install it on the server.
2. Run the following script:

```
perl inventory_export.pl -ip Ip Address -u user -p password -f output
file
```

where,

- -ip—gateway IP address
- -u—username
- -p—password
- -f—output file name (default is output.xls)

You can run the script in one of the following modes:

- d—debug mode

- s—silent mode (no logger messages)

The output file is placed under the same directory as the script.

3. To open the report, transfer the output file by FTP back to your PC, where Office Excel is installed.

8.4 Appendix C, "Change the Root-Cause Analysis Mechanism"

These topics describe the Drools Rules engine and its usage in Cisco Prime Network:

Note Contact a developer support representative if you want to add a new Drools Rule to manage and troubleshoot the fault in your network. For a list of class associations to help you select the condition to be included in the rule, see the IMO documentation available at *https://developer.cisco.com/site/prime-network/*.

Additional information about Drools Rules is available on the Cisco Developer Network (CDN).

8.4.1 What Is the Drools Rules Engine

The Drools Rules engine is a general-purpose expert system generator that combines rule-based techniques and object-oriented programming. Drools implements and extends the Rete algorithm, and uses the rule based approach to implement an expert system. Drools is based on an object-oriented paradigm and uses user-defined rules to perform pattern matching on different conditions. It also provides a customizable mechanism to add decision support and data-flow control functions to business applications.

The Drools syntax is XML-based with embedded Java code, and is organized into source files (known as a rule files), which are plain ASCII files. Drools uses objects as marked by patterns and rules that invoke certain actions.
Additional notes:

- Drools objects are Java objects and can be represented by instances of Java classes or XML schemas.

- A pattern is a coded expression (program) that manipulates one or more objects to form a pattern to make, adapt, or fashion behavior according to designed logic.
- Working memory is where Drools stores all objects that it is handling.
- Actions are operations that might change the working memory.
- A rule can perform many types of actions, such as, executing a method on one of the objects.
- Drools stores in the agenda the list of rules to be executed.

For more information about Drools Rules see www.jboss.org/drools/

Drools Rules and Prime Network

Prime Network uses Drools version 2.1. The Drools Rules engine enables you to extend the Prime Network alarm correlation mechanism with user-defined rules and business logic. It can be used to enable business rules to be applied on event updates.

Drools Rules can be triggered on tickets, network events, and non-network events.
Rules are applied after new ticket is initiated or a new event (network or non-network event) is initiated in Prime Network. Non-actionable events are not supported by the Drools Rules engine.

The Drools Rules engine is fully integrated within the Prime Network gateway and does not require any synchronization or maintenance.
A primary advantage of the Prime Network Drools Rules engine is that it lets you act upon certain events or tickets, and fully customizes the action to the case. Here are some examples:

- Notify a network operator by e-mail when various network faults occur.
- Notify a network administrator using SMS in the case of a major network fault.
- Notify the Prime Network system administrator when system events occur, such as running out of disk space. In addition, run a disk cleaning script.
- Notify the security administrator when security events occur.

Note Some of these examples, such as sending SMS or e-mail, require interaction with third party systems not provided with Prime Network.

Drools rules can be added to Prime Network by creating new rule definitions in the ~/Main/data/post.drl file located on the Prime Network gateway server. The rules are written in XML format with embedded Java code. There is no need to compile the rules. Each rule contains a set of conditions and is executed only if the condition is true. Drools rules contain an XML entry similar to the following, which specifies whether or not they are enabled:

```
<java:condition>true</java:condition>
```

All enabled rules are loaded by the Prime Network gateway and are triggered on the relevant Prime Network objects.

Consider the following points before you build and test the rules:

- Error and processing messages are written to ~/Main/logs/11.out on the Prime Network gateway server.
- We recommend that you use this capability with some caution, as the more rules that are defined, the greater the performance impact on the system.

8.4.2 Drools Rules Definitions in Prime Network

Drools supports ticket and event IMO manipulation for each ticket/event processed by the Prime Network gateway. Prime Network maintains the following Drools processing instance (context) for the rules file:

- Postcorrelation context—Defined in the post.drl rule file. This rule is executed after processing in the Prime Network gateway.

The rule file is located under ~/Main/data in the Prime Network gateway server.
The following Drools Rules parameters are stored in the registry file mmvm.xml under the eventmanagement key:

- ContextId—The Drools context name.
- ruleFilePath—The name of the respective rule file. The specified rule (the filename under mmvm.xml).

The mmvm.xml file is located under ~/Main/registry/ConfigurationFiles/127.0.0.1.

8.4.3 Enable and Disable Drools Rules

In Prime Network, the Drools Rules engine is disabled by default. If Drools Rules are to be used, the engine should be explicitly enabled.

To enable the Drools Rules engine:

1. From ~/Main, enter the following commands:
   - ./runRegTool.sh -gs 127.0.0.1 set 0.0.0.0 site/plugin/droolsplugin/enable true
   - ./runRegTool.sh -gs 127.0.0.1 add 0.0.0.0 site/plugin/AlarmPlugin/NotificationHandlers/com.sheer.metromission.plugin.alarm.DroolsNotificationHandler
2. Restart AVM11 to apply this change.

If no rules are in use, we recommend that you disable the Drools Rules engine.

To disable the Drools Rules engine, do the following:

1. From ~/Main, enter the following commands:
   ./runRegTool.sh -gs 127.0.0.1 set 0.0.0.0 site/plugin/droolsplugin/enable false

./runRegTool.sh -gs 127.0.0.1 remove 0.0.0.0 site/plugin/AlarmPlugin/NotificationHandlers/com.sheer.metromission.plugin.alarm.DroolsNotificationHandler

2. Restart AVM11 to apply this change.

8.4.4 Modify a Rule

You can modify a rule. After you edit a rule, you need to reload the rule. To reload the rule, run the following BQL command:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="ReloadRules">
      <param name="contextId">
      <value>POST</value>
      </param>
</command>
```

8.4.5 Display Existing Drools Rules

You can check the supported Prime Network postcorrelation rules by using the following BQL command:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="GetAllRules">
      <param name="contextId">
      <value>POST</value>
      </param>
</command>
```

8.4.6 Upgrade and Validate Drools Rules Files

To upgrade a rule file:

1. Make a copy of the post.drl file, and edit it.
2. Copy the updated file (under a temporary name) to the gateway (directory $ANAHOME/Main/data).

Note If the rule file is edited on a PC, make sure that the text format is compliant to the UNIX version that runs on the gateway. If necessary, use a utility such as DOS2UNIX for the conversion.

3. Check the validity of the new file by running the following BQL command:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="ValidateRules">
      <param name="fileName">
      <value>rule-file-name-full-path</value>
      </param>
```

```
</command>
```

For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="ValidateRules">
      <param name="fileName">
      <value>/export/home/network38/Main/data/post.drl</value>
      </param>
</command>
```

4. Once the rule file has been validated:
   - Copy the new rule file over the existing rule file.
   - Reload rules files by running the following BQL command:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="ReloadRules">
  <param name="contextId">
  <value>POST</value>
  </param>
</command>
```

8.4.7 Drools Rules Examples

These topics provide examples that illustrate how Drools Rules can be used in Cisco Prime Network to enable alarm forwarding using e-mails:

- Example: Generate E-mail Upon Link Down/Up Ticket, page 435
- Example: Generate E-mail for Critical Ticket, page 436

8.4.7.1 Example: Generate E-mail Upon Link Down/Up Ticket

This example is a of a Drools Rule that generates an e-mail if a Link Down / Link Up ticket is raised.

```
<rule name="Severity critical Alert">
  <parameter identifier="networkEvent">
        <class>com.sheer.imo.newalarm.INetworkEvent</class>
  </parameter>
  <java:condition>networkEvent.getName() == 1</java:condition>
  <java:consequence>
        import javax.mail.*;
        import javax.mail.internet.*;
        import java.util.*;
        import com.sheer.system.os.interfaces.Logger;
        Logger.getInstance().fatal("\nCritical Ticket Rule Fired\n");
        String recipients[ ] = new String[] {"<emailID1>", "<emailID2>"}
   ;
        String subject = networkEvent.getDescription();
        String message = "A Link Down/up Ticket has been raised. Please
        attend to the problem immediately";
        String from = "CiscoANA@company.com";
        boolean debug = false;
```

```
        //Set the host smtp address
        Properties props = new Properties();
        props.put("mail.smtp.host", "<smpt server name>");
        // create some properties and get the default Session
        Session session = Session.getDefaultInstance(props, null);
        session.setDebug(debug);
        // create a message
        Message msg = new MimeMessage(session);
        // set the from and to address
        InternetAddress addressFrom = new InternetAddress(from);
        msg.setFrom(addressFrom);
        InternetAddress[] addressTo = new
        InternetAddress[recipients.length];
        for (int i = 0; i &lt; recipients.length; i++) {
        addressTo[i] = new InternetAddress(recipients[i]);
        }
        msg.setRecipients(Message.RecipientType.TO, addressTo);
        // Setting the Subject and Content Type
        msg.setSubject(subject);
        msg.setContent(message, "text/plain");
        Transport.send(msg);
  </java:consequence>
</rule>
```

8.4.7.2 Example: Generate E-mail for Critical Ticket

The following Drools Rule example generates an e-mail if a Ticket with severity Critical has been raised.

Note Only new ticket notifications will be captured here—updates on this ticket will not be received. For example, if a ticket is initiated with Aggregated Severity MINOR and then due to correlation with a critical alarm the severity was changed to CRITICAL, it won't be captured in this rule.

```
 <rule name="Severity critical Alert">
     <parameter identifier="ticket">
     <class>com.sheer.imo.newalarm.ITicket</class>
     </parameter>
   <java:condition>ticket.getAggregatedSeverityEnum() ==
6</java:condition>
     <java:consequence>
     import javax.mail.*;
     import javax.mail.internet.*;
     import java.util.*;
     import com.sheer.system.os.interfaces.Logger;
     import com.sheer.imo.newalarm.ITicket;
     import com.sheer.imo.keys.INewAlarmOid;
     import com.sheer.client.common.util.DisplayNameFactory;
     Logger.getInstance().fatal("\nCritical Ticket Rule Fired\n");
     String recipients[ ] = new String[] {"<emailID1>", "<emailID2>"} ;
     String subject = ticket.getDescription();
     String message = "Ticket ID = " +
     ((INewAlarmOid)ticket.getObjectId()).getId() + "\n"            +
     "Severity = " +
     ITicket.S_AGGREGATED_SEVERITY[ticket.getAggregatedSeverityEnum()]+
     "\n" + "Creation Time = " + ticket.getCreationTime()
```

```
    + "\n" + " Location =" +
    DisplayNameFactory.getTextForIOid(ticket.getSource());
    String from = "CiscoANA@company.com";
    boolean debug = false;
    //Set the host smtp address
    Properties props = new Properties();
    props.put("mail.smtp.host", "<smpt server name>");
    // create some properties and get the default Session
    Session session = Session.getDefaultInstance(props, null);
    session.setDebug(debug);
    // create a message
    Message msg = new MimeMessage(session);
    // set the from and to address
    InternetAddress addressFrom = new InternetAddress(from);
    msg.setFrom(addressFrom);
    InternetAddress[] addressTo = new
    InternetAddress[recipients.length];
    for (int i = 0; i &lt; recipients.length; i++) {
    addressTo[i] = new InternetAddress(recipients[i]);
    }
    msg.setRecipients(Message.RecipientType.TO, addressTo);
    // Setting the Subject and Content Type
    msg.setSubject(subject);
    msg.setContent(message, "text/plain");
    Transport.send(msg);
  </java:consequence>
</rule>
```

Note To create this rule for a MAJOR ticket, change the <java:condition> section as follows:

```
<java:condition>((ticket.getAggregatedSeverityEnum() ==
6)||(ticket.getAggregatedSeverityEnum() ==
5))</java:condition>
```

Note The Properties object in the above examples is used only the first time Session.getDefaultInstance (props, null) method is called, when a new Session object is created. Subsequent calls return the Session object that was created by the first call, and ignore the passed Properties object. Use the Session.getInstance (props, null) method to get a new Session object every time the method is called (for example, if you need to change the Mail SMTP Host).

The following table describes the properties used in the above examples.

Property	Description
rule name	The name of the rule.
parameter identifier	A definition of a parameter to use in this rule context. The parameter value will be any instance of the class defined within the <class></class> brackets.

Property	Description
java:condition	The condition that initiates the e-mail generation, based on the rule definitions. For Example 1, the condition is that a Link Down/Up ticket is raised. For Example 2, the condition is that a ticket with Critical severity is raised.
java:consequence	The set of Java operations to run in case the previous conditions have been met. This section should include your drool rule operations.
Logger.getInstance	Helps to identify whether a rule was triggered. After you reload the rule, go to ~/Main/logs directory and enter tail -f 11.out. Check the log file to find out whether the rule was triggered.
String recipients	The e-mail IDs to which you want the system to send an e-mail.
String subject	The string to fetch the description for the alarm / ticket. In example 1, the string is networkEvent.getDescription().
String from	The e-mail ID to be displayed in the From field.

9 Index