Oracle® Communications IP Service Activator

Juniper M-series Device Support Guide

Release 7.2

E47718-01

October 2013

ORACLE®

Oracle Communications IP Service Activator Juniper M-series Device Support Guide, Release 7.2

E47718-01

Contents

4 Manual Preconfiguration

5 Configuration of MPLS VPNs

6 Configuring Layer 2 Martini VPNs

7 Configuring QoS and Access Control Features

8 Troubleshooting

9 Useful References

A MPLS-VPN Device Configuration

Preface

The Juniper M-series Device Support Guide provides detailed technical information about the Juniper M-series Device Driver, including supported features, configuration requirements, and detailed examples.

Audience

This guide is intended for network managers responsible for implementing Oracle Communications IP Service Activator within a network using Juniper M-series and T-series routers.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

1

Summary of Supported Features

This chapter provides an overview of Oracle Communications IP Service Activator support for Juniper M-series and T-series devices.

Supported IP Service Activator Features

This section provides an overview of how the various features and configurations modelled in IP Service Activator are mapped to and supported on Juniper hardware.

IP Service Activator supports the provisioning of a rich set of advanced features on Juniper hardware including QoS, a wide range of tunnelling protocols (MPLS VPNs, Layer 2 Martini VPNs, and MPLS LSPs), and routing protocols including eBGP, OSPF, RIP and static routing.

Note: The provisioning and configuration features supported by IP Service Activator require the Internet Processor II ASIC. This is standard on all Juniper models except early versions of M20 and M40 devices, where it is optional.

The Juniper M-series Device Driver also supports the Configuration Template Module (CTM) which helps you to streamline the activation of services on network objects through the use of pre-defined or customized templates. For details, see the Configuration Template Module online Help.

QoS and Access Control Support

Table 1–1 lists the QoS and access control features that IP Service Activator supports on Juniper devices. For full details about these features and the way in which they are implemented, see Chapter 7, "Configuring QoS and Access Control Features".

Table 1–1 Supported QoS and Access Control Features

Type	IP Service Activator Feature	Internet Processor II
Policy rule	Access rules	Y
Policy rule	Classification rules	N
Policy rule	Policing rules	N
Marking	DiffServ codepoints (0-63)	N
Marking	IPv4 Precedence field	N
Marking	IPv4 ToS bits	N

Table 1–1 (Cont.) Supported QoS and Access Control Features

Type	IP Service Activator Feature	Internet Processor II
Marking	MPLS experimental bits	N
Marking	MPLS Topmost experimental bits	N
Traffic classification	IP Address (source/destination)	Y
Traffic classification	IP Port (source/destination)	Y
Traffic classification	IP Protocol	Y
Traffic classification	Packet marking traffic type	Y
Traffic classification	URL traffic type	N
Traffic classification	MIME traffic type	N
Traffic classification	Application protocol traffic type	N
Traffic classification	Domain Name traffic type	N
PHB groups	Rate Limiting	Y
PHB groups	WRED	N
PHB groups	WFQ	N
PHB groups	WRR	Y
PHB groups	Priority Queuing	N
PHB groups	ATM Traffic Shaping	N
PHB groups	FRTS Traffic Shaping	N
PHB groups	MQC	N

MPLS VPN Support

Table 1–2 lists the MPLS VPN features that IP Service Activator supports on Juniper devices. For full details about these features and the way in which they are implemented, see Chapter 5, "Configuration of MPLS VPNs".

Table 1–2 Supported MPLS VPN Features

Type	IP Service Activator Feature	Internet Processor II
VRF table	User-defined VRF table name	Y
VRF table	VRF re-use/reduction	Y
VRF table	User-defined RD numbers	Y
VRF table	User-defined RT numbers	Y
VRF table	RDs per VPN	Y
VRF table	VRF route limit (max routes	Y
VRF table	Co-existence with pre-defined VRFs	Y
VRF table	Pre-defined export maps	Y
PE-PE peering (iBGP)	iBGP peering optional	Y
PE-PE peering (iBGP)	Maximum paths	Y

Table 1–2 (Cont.) Supported MPLS VPN Features

Type	IP Service Activator Feature	Internet Processor II
PE-PE peering (iBGP)	Extended/standard community attributes (This feature is not available on Juniper M-series or T-series devices.)	N
PE-PE peering (iBGP)	PE-PE MD5 authentication	Y
PE to CE connectivity	eBGP	Y
PE to CE connectivity	OSPF	Y
PE to CE connectivity	RIP	Y
PE to CE connectivity	Static routing	Y
eBGP configuration	AS override	Y
eBGP configuration	Allow AS in	Y
eBGP configuration	Extended/standard community attributes (This feature is not available on Juniper M-series or T-series devices.)	N
eBGP configuration	Local preference	Y
eBGP configuration	PE-CE authentication	Y
eBGP configuration	Prefix filters	Y
eBGP configuration	Prefix limit	Y
eBGP configuration	Site of origin	Y
eBGP configuration	Multi-path load sharing	Y
eBGP configuration	Route dampening	Y
eBGP configuration	Route redistribution into eBGP	Y
OSPF	Route redistribution into OSPF	Y
OSPF	Domain Tag	Y
RIP	Route redistribution into RIP	Y
Static configuration	Global routes	Y
Static configuration	Permanent routes	Y

Layer 2 Martini VPN Support

Table 1–3 lists the Layer 2 Martini VPN features that IP Service Activator supports on Juniper devices. For full details about these features and the way in which they are implemented, see Chapter 6, "Configuring Layer 2 Martini VPNs".

Table 1–3 Supported Layer 2 Martini VPN Features

Type	IP Service Activator Feature	Internet Processor II
Layer 2 Martini VPN	Layer 2 Martini VPN	Y
Layer 2 Martini VPN	Provisioned sub-interfaces	Y
Layer 2 Martini VPN	Virtual Circuit ID	Y
Encapsulation	Frame Relay ATM AAL5	Y

Table 1–3 (Cont.) Supported Layer 2 Martini VPN Features

Type	IP Service Activator Feature	Internet Processor II
Encapsulation	Frame Relay ATM Cell	Y
Encapsulation	Frame Relay	Y
Encapsulation	Ethernet	Y
Encapsulation	Ethernet VLAN	Y

MPLS Label Switched Path (LSP) Support

Table 1–4 lists the MPLS LSP features that IP Service Activator supports on Juniper devices.

Table 1–4 Supported MPLS LSP Features

Type	IP Service Activator Feature	Internet Processor II
Tunnel	Hold and Setup Priority	Y
Tunnel	Admin Groups (Coloring)	Y
Tunnel	IGP Metric	Y
Tunnel	LDP	Y
Protection	Fast Re-Route	Y
Protection	Node and Link Protection	Y
Paths	Primary and Secondary Paths	Y
Paths	Next Hop Lists	Y
Paths	Exclude Address Lists	Y
Paths	Dynamic and Explicit Paths	Y

Supported Juniper M-series and T-series Hardware and Software

The Juniper M-series Device Driver is effectively capable of configuring any JUNOS-based device. The exact capabilities that can be supported by IP Service Activator depend on the device model, the operating system that it is running and the interface.

If you are using, or wish to use, different hardware or software from that defined here, please contact Global Customer Care.

Supported Devices

The Juniper M-series Device Driver supports Junos-based M- and T-series devices.

Supported JUNOS Versions

Refer to *IP Service Activator Release Notes* for your IP Service Activator software release for support information for the specific releases of the JUNOS operating system.

Juniper-supported Interface Types

Juniper devices support a wide range of interfaces. The Juniper device driver supports a subset of these interfaces. For information on Juniper interface types available, consult the Juniper website:

http://www.juniper.net

Table 1–5 summarizes support for interface types on those Juniper M-series and T-series devices that are supported by IP Service Activator.

Table 1–5 Interface Types Supported on Juniper M-series and T-series Devices

Supported Juniper Interface Configuration Prefix	Physical Interface
ae	aggregated Ethernet
as	aggregated Sonet/SDH
at	ATM
ds	DS0
e1	E1
e3	E3
fe	Fast Ethernet
fx	management internal ethernet interfaces
ge	Gigabit Ethernet
gr	Generic Route Encapsulation (GRE) tunnel
ip	IP-overIP encapsulation tunnel
lo	Loopback
ml	Multilink
mt	Multicast tunnel
so	SONET/SDH
t1	T1
t3	T3
vt	Virtual loopback tunnel

Supported Traffic Types

Table 1–6 lists the traffic classification types that are supported on Juniper M-series devices.

Table 1–6 Traffic Types Supported on Juniper M-series Devices

Traffic Type	Supported on Juniper M-series
Port/IP protocol	Y
DiffServ codepoint	Y
IP Precedence	N
MPLS experimental bits	N
Domain	N

Table 1–6 (Cont.) Traffic Types Supported on Juniper M-series Devices

Traffic Type	Supported on Juniper M-series
Application	N
Sub-application	N
URL	N
MIME	N

Enhanced Command Threshold Control for Juniper Device Driver

The Juniper Device Driver (JDD) command threshold is used to monitor configuration differences between routing-instance and interface counts. It was previously able to address only one criteria at a time, whereby the user was unable to select routing-instance and interface at the same time.

To address this issue, JDD configuration thresholding has been enhanced to monitor routing-instance deletions that affect a certain number of interfaces, and block the configuration based on that specific number of interfaces crossing the threshold value for service-instance, on a per service-instance basis.

The enhanced command threshold control in JDD is to extend the logic for the routing-instance. The Command Threshold configured in the device properties is checked against each of the following criteria:

- Number of routing-instances that are getting deleted.
- Number of interfaces linked to a routing-instance that is getting deleted.

Note: The user must check every routing-instance that is getting deleted one by one.

If one of the above numbers exceeds the threshold, then the changes must be blocked. This feature is supported only on Juniper Device Driver.

2

Installation and Set-up Notes

This chapter contains important notes on setting up the Juniper M-series Device Driver.

For full details about installing Oracle Communications IP Service Activator, including the Juniper M-series Device Driver, refer to *IP Service Activator Installation Guide*.

Installation Notes

The Juniper M-series Device Driver is a separate executable component called **juniper_device_driver**.

The Juniper M-series Device Driver is installed by default when you select the Proxy Agent component for installation.

Command-line Parameters

Table 2–1 summarizes available command-line parameters. For more details of the options for off-line configuration, see "Lengthening the Time Between IP Service Activator Commit Commands".

Table 2–1 Command-line Parameters

Parameter	Description
-ComponentName *name*	Specifies the name of the Juniper M-series Device Driver component as displayed in the client.
-ComponentLocation *hostname*	Specifies the hostname on which the Juniper M-series Device Driver component is installed.
-ConnectTimeout *n*	Socket connection timeout, where *n* is an integer specifying the number of seconds. Default is 30 seconds.
-LineDelay *milliseconds*	Amount of time, in milliseconds, that IP Service Activator waits between delivering lines of configuration to the device. **Note**: This parameter is only applicable if you access the device through Telnet, and not SSH.

Table 2–1 (Cont.) Command-line Parameters

Parameter	Description
-NoSystemGroupAddresses	If this flag is specified, VPN site addresses are specified in the orchestream configuration group instead of the system configuration group. Use this option if you need to manually provision additional addresses on an interface that will co-exist with IP Service Activator managed addresses on that interface. **Note**: Ensure that manually configured addresses in the system group do not conflict with IP Service Activator configured addresses in the orchestream group.
-OverwriteInterfaceDescription	The interface description configured in the Site Properties dialog is enabled only when the device driver is started with this option.
-ReadTimeout *n*	Socket read timeout in seconds, where *n* is an integer specifying number of seconds. Default is 30 seconds.
-UseRIBGroup	Use rib-group definitions statements when provisioning routing instances in MPLS VPNs. This overrides the use of policy-based mechanisms such as the auto-export JUNOS command.
-WriteTimeout *n*	Socket write timeout in seconds, where *n* is an integer specifying number of seconds. Default is 30 seconds.
-FileConfigDelivery	With this command enabled the device driver does not attempt to enter configure mode on the device and does not apply any changes to the FileInterface file. Instead, all configuration commands are delivered to a file named **orchestream-config-***device_name-time-date***.txt**. This file is created in a directory specified by -FileConfigDeliveryDir
-FileConfigDeliveryDir *name*	Specifies the location of the **orchestream-config-***device_name-time-date***.txt** file.
-NoDeviceComm	If this flag is specified, both CLI and SNMP connections are disabled, preventing the device driver connecting to the device.
-UseMaxPrefixes	Supports maximum prefixes from JDD when a new command line option -UseMaxPrefixes is passed to JDD.This option can also be set from ComponentParametes.By default this flag is off and not passed as cmd line option. If it is passed or set to true from ComponentParameters, JDD will push maximum-prefixes instead maximum-routes.

Note: For delivery of configuration to file instead of the device, the following commands must be configured:

- -FileConfigDelivery
- -FileConfigDeliveryDir
- -NoDeviceComm

There are also command-line parameters that control debugging logs for all IP Service Activator components. These are described in full in *IP Service Activator Administrator's Guide*.

The device driver must be restarted for any changes to these command-line parameters to take effect.

Setting Command-line Parameters on Oracle Solaris

Command-line options are specified in the **cman.cfg** file located in the **/opt/OracleCommunications/Service Activator/config** directory.

Using a text editor such as vi, edit the juniper entry in the **cman.cfg** file with the relevant option.

Exporting Routes Between Routing Instances in Overlapping MPLS VPNs

Starting with IP Service Activator version 4.2, the JUNOS auto-export command is used by default to implement policy-based export of routes between routing instances in overlapping MPLS VPNs. This greatly simplifies the device configuration required to set up overlapping VPNs.

The Juniper Device Driver and Existing RIB-group Statements

If you have upgraded to IP Service Activator version 4.2 or higher and your devices have RIB groups defined, or if you have been using the -UseRIBGroup command-line parameter (see Table 2–1) and have removed it, the Juniper device driver handles existing configuration in the following way. It will:

- Delete rib-group definition statements inside group orchestream
- For each routing instance inside group orchestream:
  - Change the routing-options to auto-export
  - Delete the interface-routes sections which defines the RIB group
- Delete the definition of the family inet which refers to the RIB group, inside the protocols definition

Because of these changes, a simple upgrade procedure is recommended. See "Upgrading Devices from Using RIB Groups to Using auto-export".

Upgrading Devices from Using RIB Groups to Using auto-export

If you have upgraded to IP Service Activator version 4.2 or higher and your devices have RIB groups defined, or if you have been using the -UseRIBGroup command-line parameter and are removing it, upgrade you devices to use auto-export.

To upgrade your devices:

1. Un-manage affected devices.
2. Perform the upgrade (or de-implement the -UseRIBGroup command-line parameter).
3. Restart IP Service Activator.
4. Manage the affected devices.

   When the devices are re-managed, the new configuration using the auto-export command is pushed.

Over-riding the auto-export Command with -UseRIBGroup

To configure the Juniper device driver to use RIB groups instead of the auto-export command to export routes in MPLS VPNS, use the -UseRIBGroup command-line parameter. This mimics the handling of export routes in versions of IP Service Activator prior to version 4.2. See "Setting Command-line Parameters on Oracle Solaris".

3

Discovery and Configuration

This chapter provides some basic information about JUNOS configuration files before describing how the Juniper M-series Device Driver (JDD) configures devices.

JUNOS Configuration Files

This section describes the JUNOS configuration files. It includes the following:

- About the Configuration Hierarchy: describes the hierarchy of the JUNOS configuration files
- About Configuration Groups: describes configuration groups and their hierarchy
- About Active Configuration: describes how the JDD activates new configurations

There are two types of statement in a JUNOS configuration file:

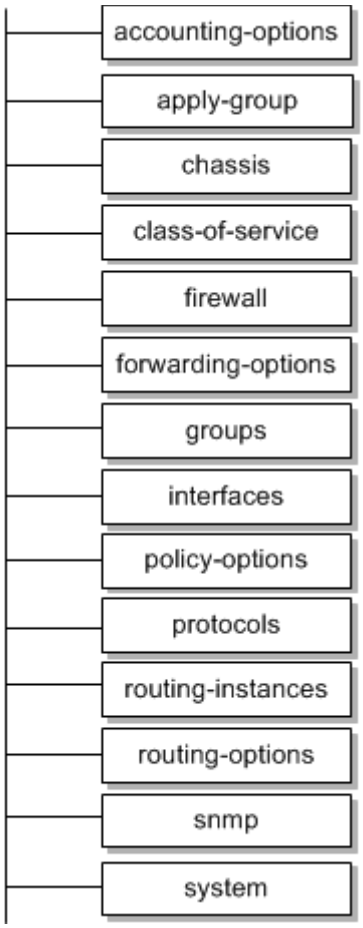
- Container statements: hold a collection of statements
- Leaf statements: sit within a container statement

Container statements may be nested.

About the Configuration Hierarchy

The container and leaf statements that are entered in the configuration file make up a configuration hierarchy. Most top-level statements are container statements that hold other container statements that form the tree branches. The leaf statements are the leaves of the hierarchy tree. Figure 3–1 shows the top-level statements in the Juniper configuration hierarchy.

Figure 3–1 Juniper Configuration Hierarchy



Within the configuration file, statements are indented to show their relative position in the hierarchy. For example:

```
firewall {
  family inet {
    filter InFilter--at-0-0-0-1 {
      term OrchFilterTerm--4688 {
        from {
          precedence flash-override;
          source-address 150.150.150.0/24;
          destination-address 10.50.0.7/32;
        }
        then {
          accept;
        }
      }
    }
  }
}
```

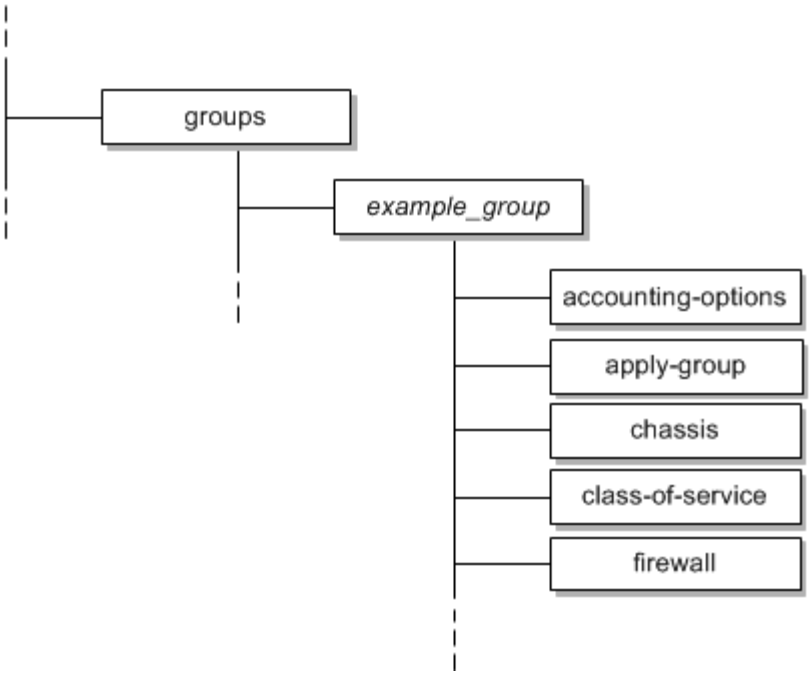
An individual hierarchy of statements, starting at the base of the hierarchy, is referred to as a statement path. For example, the statement path for the above example is expressed as **[edit firewall]**.

About Configuration Groups

Statements within the JUNOS configuration file may be collected in a named group that can be applied to the rest of the configuration. Any number of groups may be defined. A group, or selected sections of the group, can be applied to different sections of the configuration file.

Configuration groups are defined at the top level of the configuration hierarchy. Within each group, the same configuration hierarchy exists as at the top level. This is shown in Figure 3–2.

Figure 3–2 Configuration Groups and Configuration Hierarchy



The Juniper M-series Device Driver uses the configuration groups feature to aid separation of Oracle Communications IP Service Activator-generated configuration. Any configuration that is placed on the device is defined within the orchestream configuration group. The following example shows the opening lines of the group definition:

```
groups {
  orchestream {
    interfaces {
      at-1/0/0 {
      …
```

This group is then applied to the main configuration by the apply-groups command.

> **Note:** When configuring an MPLS VPN, if the IP address specified for an interface through IP Service Activator differs from the currently-configured values on the device, IP Service Activator changes the address within the main configuration. The change is commented. This change is made in the main configuration because any interface addresses defined in the orchestream group are merged with those in the main configuration.

About Active Configuration

JUNOS holds ten versions of the configuration file, one of which is the currently running configuration. In Juniper terminology, this is the active configuration. This configuration may be edited to produce a new configuration which is referred to as the candidate configuration.

When a JUNOS **commit** command is run, the candidate configuration becomes the active configuration file 0). The active configuration becomes the first backup (configuration file 1), and so on.

The Juniper M-series Device Driver uses the **commit** command when pushing new configuration to the device.

If the device driver is configured for **no commit**, it uses **commit confirm** and if successful, it saves the configuration on the device with the name **orchestream_new_config.txt**. You may then apply this configuration manually if you choose.

If the driver is configured for normal operation, it uses **commit confirmed 1**. In this case, device commits the configuration and then waits for the Juniper Device Driver to send a **Y** one second later. If the device does not receive the **Y** response, it assumes that connectivity to the management station has been lost and rolls back the configuration. This is very useful to ensure that you are not locked out of the device by an errant configuration.

Communication and Authentication

The Juniper M-series Device Driver accesses devices by the command-line interface (CLI). Access is authenticated by Tacacs+, Telnet (Named User), or SSH/RSA with password authentication. You must ensure that the authentication methods are correctly set up for all Juniper devices in your network. You can set the authentication on the Security page in the Discovery dialog box to ensure that it applies to all devices, or set it for individual devices.

Note: The user specified on the Security page must have the appropriate privilege level on each PE device to be managed. For more information, see "Configuring User Access Privileges".

The Juniper M-series Device Driver requires SNMP to be enabled on devices for the IP Service Activator discovery process to work and so that the capabilities of Juniper routers can be obtained. Ensure that SNMP is set up correctly. For more information, see "Configuring SNMP".

Configuring SSH Authentication Keys

To support SSH with password authentication on Juniper M-series devices, configure the SSH authentication keys when the new IP Service Activator build is installed.

To configure the SSH authentication keys:

1. Enable SSH service on the devices.
2. On the IP Service Activator installation machine, ensure that the directory *ipsaadmin_home_directory***/.ssh** is empty. The default *ipsaadmin_home_directory* directory is **/export/home/ipsaadmin**.
3. Generate a new key pair using the key generator located in the *Service_Activator_Install***/ssh/bin** directory (typically

opt/OracleCommunications/ServiceActivator/ssh/bin) using the following command:

```
ssh-keygen -f id_rsa -t rsa -b 1024
```

When prompted for a key password by the key generator, leave the passphrase blank.

Two files are generated:

- **id_rsa**
- **id_rsa.pub**

4. Copy **id_rsa** and **id_rsa.pub** to *ipsaadmin_home_directory***/.ssh**.
5. Set the protections of the files using the following command:

```
chmod 600 id_*
```

When setting the security preferences for the device/domain, select **SSH with password authentication** and enter the valid username and password for the router.

Note: The keyed authentication option is not supported.

Obtaining Device Capabilities

The features supported on each device are dependent on the hardware and software combination and are indicated by a list of capabilities returned from each device. You should always check capabilities before configuring network devices. The Juniper M-series Device Driver does not need correct security settings in order to obtain capabilities.

The Juniper device driver uses SNMP to retrieve the device capabilities.

Capabilities are obtained at two levels:

- At device level, the capabilities specify the device's support for SLA measurement techniques, Transparent LAN Services and VLANs. These features are not currently supported by IP Service Activator on Juniper M-series or T-series devices
- At interface, sub-interface (unit) and VC endpoint level, the capabilities indicate the QoS, MPLS, and Layer 2 Martini VPN features that can be configured on that interface

The device driver attempts to obtain the capabilities automatically at the end of the discovery process. If this fails you can initiate the process to fetch capabilities manually.

If IP Service Activator is unable to discover capabilities during the discovery process then you can request the capabilities at a later point.

In addition, if the device capabilities have changed, for example, as a result of an operating system upgrade, you can reset and re-fetch capabilities after ensuring the device is unmanaged.

Note: The provisioning and configuration features supported by IP Service Activator require the Internet Processor II ASIC. The Juniper M-series Device Driver is optimized to correctly report capabilities from Juniper devices with this version of the ASIC. With earlier ASIC versions, support for rate limiting and access rules is incorrectly reported.

There may be inconsistencies discovering pre-configured DLCIs, VLANs and VCs. See "Working with Pre-existing VLAN, DLCI, and VC Endpoints" for details.

For more details on the discovery process, see *IP Service Activator User's Guide*.

Applying IP Service Activator Configuration

The M-series device driver maintains a virtual version of the device state, consisting of the rules, PHB groups, and so on that have been defined in the IP Service Activator-generated configuration group. The device maintains the real state in the active configuration.

When configuration is pushed to the device, the Juniper device driver performs the following steps:

1. The driver logs into the device and obtains the current configuration.
2. The driver enters configuration mode by running the edit exclusive command.
3. If another entity is editing the configuration, no new IP Service Activator configuration is installed on the device. An error message is logged and displayed in the IP Service Activator client.
4. If no entities are editing the configuration, the driver edits the candidate configuration. The configuration file is locked to all other entities. When configuration is complete, the commit check command is run and JUNOS verifies the candidate configuration:
5. If the command succeeds, the candidate configuration is made active by sending a commit command.
6. If JUNOS finds an error, an error is raised against the device. No new IP Service Activator configuration is installed. The device driver saves the failed configuration to a file named **last_failed_orchestream_config** on the Juniper device. This log file contains details about the error and can be reloaded in order to diagnose and correct the problem.

Setting Routing Instance Annotation Limits

You can set the maximum limit for routing instance annotation. Issues that are related to long routing instance descriptions on Juniper routers can cause configuration problems. To avoid this, you can set the AnnotationLineLimit using either the command line option or component parameters.

To set AnnotationLineLimit using the command line:

1. Enter the following command:

```
-AnnotationLineLimit n
```

where *n* is an integer. Setting *n* to -1 retains the default behavior and the annotation line is not split. Setting *n* to 0, removes the annotation line. Setting *n* to a number greater than 0 splits the annotation line at *n* characters.

For example, if the annotation line has 40 characters, and the annotation line limit is set to 20, the annotation line is split into two lines.

In the following example, the annotation line is a maximum of 20 characters, including "/*" and "*/".

```
Input String:      S2 + S3 + S1
       Output Annotate Line:
          /* S2 + */
          /*  S3  */
          /* + S1 */
The command line option is -AnnotationLineLimit 40
```

To set AnnotationLineLimit using the component parameters:

1. Use the following component parameter:

   ```
   ComponentParameters -ComponentName  juniper-shmitra-ca -set
   -AnnotationLineLimit n.
   ```

   where *n* is an integer. Setting *n* to -1 retains the default behavior and the annotation line is not split. Setting *n* to 0, removes the annotation line. Setting *n* to a number greater than 0 splits the annotation line at *n* characters.

Checking and Forcing Consistency

A check and force consistency process ensures that the configuration of each device always matches the virtual device state. A device may lose its configuration if it goes down. The process works as follows:

1. The proxy agent regularly polls the UpTime MIB objects which represent the devices that the driver controls. If it finds that the device was down (that is, it has re-booted since the last time it was checked), the proxy agent tells the driver to check the consistency of the device configuration.
2. The driver initiates the sequence outlined in "Applying IP Service Activator Configuration".

Maintaining Synchronization Between Routing Engines

When Routing Engine redundancy is implemented, the configuration pushed to the device by IP Service Activator will only be updated on the Master Routing Engine. The Backup Routing Engine will not be updated with the configuration. As a result, the Backup Routing Engine will not be correct if there is a need to switch over to this engine.

A script is available which maintains synchronization between the command caches of both engines. This script will issue a commit synchronize command to update the Backup Routing Engine and synchronize with the Master Routing Engine after every successful configuration change.

The script file can be found at the following location:

/opt/OracleCommunications/Service Activator/DriverScripts/juniper/CommitSynchronize.py

See the IP Service Activator online Help for information on how to import a driver script file.

Lengthening the Time Between IP Service Activator Commit Commands

Issues related to commit timing on Juniper routers cause configuration rollback. To address this problem, Juniper Device Driver has been enhanced so that the user can set the commit confirm time and commit delay time through the command line option or component parameters CommitConfirmTime and CommitDelayTime.

The unit for CommitConfirmTime is minute and the unit for CommitDelayTime is second.

To set commit confirm time and commit delay time using the command line:

1. Enter the following commands:

```
-CommitConfirmTime n
-CommitDelayTime m
```

   where *n* and *m* are integers.

   For example, the following commands set the commit confirm time to five minutes and the commit delay time to three seconds:

```
-CommitConfirmTime 5
-CommitDelayTime 3
```

To set commit confirm time and commit delay time using the component parameters:

1. Use the following component parameters:

```
ComponentParameters -ComponentName juniper-Marina-ca -set -CommitConfirmTime n
ComponentParameters -ComponentName juniper-Marina-ca -set -CommitDelayTime m
```

   where *n* and *m* are integers.

The Juniper python script **Commit Synchronize Juniper Driver Script** has also been enhanced so that the user can set the synchronizing delay time.

The unit for COMMIT_SYNC_DELAY . COMMIT_SYNC_DELAY is second.

To set the synchronizing delay time in the python script:

1. Replace 0 as the value for COMMIT_SYNC_DELAY in the following install section of the python script with the desired delay time:

```
commit synchronize:
  COMMIT_SYNC_DELAY = 0
  try :
  _device.openSession()

  _device.deliverCommand('configure')
   time.sleep(COMMIT_SYNC_DELAY)
  _device.deliverCommand('commit synchronize')
  _device.deliverCommand('exit')

  _device.closeSession()

  except :

  _result.setCode(_result.FAILED)
   _result.setDetails('Failed to issue command: commit synchronize')

  else :

   _result.setCode(_result.OK)
  _result.setDetails ('Succeeded to issue command: commit synchronize')
```

Offline Configuration

You can save the configuration output from the device driver to a file instead of a device. The following options are available:

- Output the orchestream configuration group to a file named **device-ip-address**, local to the Juniper device driver. Output is in XML format, with the output configuration enclosed by markup tags.
- Output the orchestream configuration group to a file named **orchestream-config-devicename-time-date-.txt**, local to the Juniper device driver. The output file includes a **replace:** tag, enabling you to load the file's configuration group into the active configuration.
- Output the orchestream configuration group to a file named **orchestream_new_config.txt** in the user's home directory on the device.
- Disable communication with the device through the CLI and SNMP when outputting configuration to the **orchestream-config-devicename-time-date.txt file**.

Oracle recommends that you use file output only if it is necessary for your environment.

Implementing Offline Configuration

To implement offline configuration:

1. In the **cman.cfg** file, insert the following options in the juniper entry:

```
- FileConfigDelivery
- FileConfigDeliveryDir /opt/OracleCommunications/ServiceActivator/WorkingData
- NoDeviceComm
```

2. Restart the component manager with the following command:

```
(ipsacm start)
```

3. Run device discovery and manage your device(s).
4. Create provisioning data and commit a transaction.

   For each subsequent committed transaction that would otherwise write directly to the device, a ready-to-load file is created in **/opt/OracleCommunications/ServiceActivator/WorkingData directory** with the following file name format:

   Orchestream_config_*DeviceName_time_date***.txt**

Command-line Parameters

Table 3–1 summarizes the command-line parameters are used to control offline configuration. They must be used together for off-line configuration to work.

Table 3–1 Offline Configuration Command Line Parameters

Parameter Name	Type	Purpose
-FileConfigDelivery	Flag	Specifies that configuration will not be pushed onto the device.
-FileConfigDeliveryDir	String	Specifies the folder where the files to which configuration will be written are located.

Table 3–1 (Cont.) Offline Configuration Command Line Parameters

Parameter Name	Type	Purpose
-NoDeviceComm	Flag	Specifies that the device will not be contacted to perform **show run** or **show vers** commands or SNMP polls.
-EnableFtp	Flag	Specifies that the Juniper Device Driver will use FTP instead of Telnet to configure the device.

The following sections give more details about each parameter.

The -FileConfigDeliver and -FileConfigDeliveryDir Parameters

When you specify the -FileConfigDelivery flag, you prevent the driver from entering configure mode on the device and applying changes to the **FileInterface** file. Instead, a file named **orchestream-config-device_name-time-date.txt** is created in a directory specified by -FileConfigDeliveryDir. You must use the -NoDeviceComm flag in conjunction with these flags, and restart the component manager to make the flags effective.

The file created contains the replacement orchestream configuration group with a **replace:** flag appearing just before the group and interface statements. Once the file is created, load it.

To load the **orchestream-config-device_name-time-date.txt** file:

1. Run the JUNOS load replace command.

   For example:

   ```
   [edit]
   user@host# load replace orchestream-config-zeus-14.58.17-2001-10-04.txt
   ```

   This overwrites the existing orchestream configuration group with the file's content without affecting any other areas of the router configuration.

2. Ensure that the apply-groups orchestream statement still exists in the router configuration.

3. Apply the apply-groups orchestream statement with the following command:

   ```
   [edit]
   user@host# set apply-groups orchestream
   [edit]
   ```

Note: IP address changes that may be required when configuring MPLS VPNs must be applied to the main router configuration and not in the orchestream configuration group.

The -NoDeviceComm Parameter

When you specify the -NoDeviceComm flag, both CLI and SNMP connections are disabled, preventing the device driver connecting to the device. If the capabilities of the device are unknown, an over-estimation of the capabilities is performed. SNMP requests are still allowed to be performed by the rest of the system. When you use this flag with -FileConfigDelivery, a good estimation of the configuration changes required are written to the local file. See the example above or "About Active Configuration" for more details.

The -EnableFtp Parameter

By default, the Juniper Device Driver uses Telnet to configure the router. However, when you specify the -EnableFtp flag in **cman.cfg** (Juniper command line), the device driver uses an FTP session to configure the device.

The default configuration is as follows:

```
juniper-coenglv0219.us.oracle.com.proxy_device_
driver@coenglv0219.us.oracle.com[7.0.0.0.0.572]: $-EnableFtp=disabled
```

To enable FTP mode using Component Parameters:

1. Enter the following command:

```
orchadm-bash3.2:$ ./ComponentParameters -ComponentLocation
coenglv0219.us.oracle.com -ComponentName juniper-coenglv0219.us.oracle.com -set
-EnableFtp enabled
```

To verify the current value of EnableFtp:

1. Enter the following command:

```
juniper-coenglv0219.us.oracle.com.proxy_device_
driver@coenglv0219.us.oracle.com[7.0.0.0.0.572]: $-EnableFtp=enabled
```

To disable FTP mode using Component Parameters:

1. Enter the following command:

```
orchadm-bash3.2:$ ./ComponentParameters -ComponentLocation
coenglv0219.us.oracle.com -ComponentName juniper-coenglv0219.us.oracle.com -set
-EnableFtp disabled
```

Important Configuration Notes

The following are important things to note for configuration:

- Changing any of the offline configuration parameters while a device is managed can cause commit failure errors.
- The value set in the Manual config field on the Domain and Device property pages, including **Warn** and **Delete**, is ignored for Juniper M-series devices. The Juniper M-series Device Driver cannot be set up to monitor for or warn when changes to device configuration are made by other users. However, IP Service Activator can co-exist with manually applied configuration.

4

Manual Preconfiguration

In order for Oracle Communications IP Service Activator to correctly manage Juniper devices through the Juniper M-series Device Driver and to configure features on them, various values must be preconfigured on the devices. This chapter describes the manual Juniper device preconfiguration steps that are required.

Configuring SNMP

SNMP must be enabled on all Juniper M-series routers for the IP Service Activator discovery process to work and so that the capabilities of Juniper routers can be obtained. Ensure the following statement is included in the configuration:

```
[edit]
snmp;
```

IP Service Activator's network discovery process uses a default community of **public**; you must amend the appropriate SNMP parameter on the client's Discovery dialog box if you set a different read community on the devices.

To configure the community string:

1. Use the following command:

   ```
   [edit snmp]
   community community-string {
     authorization read-only;
     clients {
       default restrict;
       address;
     }
   }
   ```

   where *community-string* is the community string, the value for authorization is the access authorization level, and *address* is an SNMP client that is authorized to access the router.

Configuring User Access Privileges

For the Juniper M-series Device Driver to configure a device, the user specified in IP Service Activator's device security settings must have access privileges that permit interface configuration, routing configuration, firewall, and firewall-control permissions.

Users are assigned access privileges by login classes. To assign the appropriate privileges to a user:

1. At the [edit system login] level, create a login class with the appropriate access level as follows:

```
class class-name {
  permissions [ access-privileges ];
}
```

   The access privileges required are: [configure edit interface interface-control routing routing-control maintenance view firewall firewall-control]

2. At the [edit system login user] level, associate the class with the relevant user as follows:

```
user user-name {
  full-name full-name;
    uid user-id;
      class class-name;
```

Enabling SSH and Telnet

Access to a device can be authenticated through Telnet or SSH.

To enable Telnet and SSH login:

1. At the [edit services] level, use the following commands:

```
services {
ssh;
telnet;
}
```

Using SSH Authentication

When using SSH, or other non-Telnet authentication, configurations are sent to the device using the **set** command. Using this notation generates a syntax error when applying communities to the import and export policy statements. Specifically, the string **community[** will fail the VPN configuration. and comment delimiters of **/* */** and **#** are not accepted.

In both cases, comment commands must done with the CLI annotate command. The annotate command is explained in the JUNOS Software CLI User Guide (v7.6 to v8.3), available on the Juniper Networks Technical Documentation Web site:

http://www.juniper.net/techpubs/en_US/release-independent/junos/information-products/pathway-pages/junos/product/

Checking for Pre-existing System and Group Configurations

Router configurations generated by the Juniper Device Driver may fail or be inactive when interfaces on the device are already configured by other users at the master system or group configuration hierarchies.

To prevent this situation, at the system or (active) groups configuration hierarchy level, check for existing configuration operating on interfaces where policy is to be applied that might potentially conflict with the planned IP Service Activator provisioning commands. In particular, check that there are no input or output firewall filters, VRFs, l2circuits, cross-connects and/or PHB Groups already configured on those interfaces.

Performing Mandatory Manual Preconfiguration for MPLS VPNs

Before using IP Service Activator to set up VPNs, some manual configuration of routers is required.

The following preconfiguration is required for each device role.

Manually Configuring PE Routers

Perform the following mandatory manual configuration on all PE (gateway) routers in the core VPN:

- Configure core interfaces to carry IP and MPLS traffic. See "Configuring IP and MPLS on Core Interfaces".

> **Note:** The Juniper M-series Device Driver assigns IP addresses and enables MPLS only on access interfaces.

- Correctly assign IP addresses.
- Set up a loopback interface. See "Configuring Loopback Interfaces".
- Implement an IGP (such as OSPF or IS-IS) in order to distribute IP routes.
- Configure LSPs between the loopback addresses of all PE devices. See "Enabling LSPs".
- Configure IBGP sessions between PE peers.

Configuring IP and MPLS on Core Interfaces

To configure IP on an interface:

1. Use the following command:

```
interface-name {
  unit logical-interface-number {
    family inet {
      address IP-address;
    }
}
```

To enable MPLS on an interface:

1. Use the following command:

```
protocols {
  mpls {
    interface {interface-name | all };
  }
}
```

Configuring Loopback Interfaces

To configure a loopback interface:

1. Use the following command:

```
interfaces {
  lo0 {
    unit 0 {
      family inet {
        loopback-address;
      }
```

```
    }
  }
}
```

Enabling LSPs

Oracle recommends that you allow LSPs to be set up as required by enabling LDP on all PE and P routers.

To enable LDP:

1. Use the following command:

```
ldp {
  interface {interface-name | all };
}
```

Manually Configuring P Routers

Perform the following mandatory manual configuration on all P (core) routers:

- Configure core interfaces to carry IP and MPLS traffic. See "Configuring IP and MPLS on Core Interfaces".
- Correctly assign IP addresses.
- Enable the protocol used to set up LSPs. See "Enabling LSPs".
- Implement an IGP (such as OSPF or IS-IS) in order to distribute IP routes.

Manually Configuring CE Routers

The CE (access) routers at customer sites are not configured to control routing by IP Service Activator, since they may not be under the control of the network service provider. Therefore they need to be manually configured.

Perform the following mandatory manual configuration:

- Configure BGP, RIP, OSPF or static routing in order to advertise reachability information between the CE and the PE.
- Set up a loopback interface on each CE router (see "Configuring IP and MPLS on Core Interfaces" and "Configuring Loopback Interfaces").

Performing Optional Manual Preconfiguration for MPLS VPNs

You can manually preconfigure routers with data which provide specific operational requirements for your MPLS VPNs. IP Service Activator is able to incorporate the following preconfigured data into the device configuration:

- Export maps, see "About Predefined Export Maps"
- Route list filters, see "About Predefined Route List Filters"
- Filtering policies for route redistribution, see "About Predefined Filtering Policies for Route Redistribution"

About Predefined Export Maps

You can manually predefine an export policy on a PE router and assign it to a VRF table in the Advanced VPN property page of the Site dialog box. The export policy only allows those routes in the VRF table whose route prefixes match those specified

in the export policy to be advertised to other PE routers. The exported routes are tagged with an RT value specified by the export map.

Using a manually predefined export policy enables you to control the redistribution of routing information into iBGP without affecting route redistribution into eBGP. This is in contrast to the route redistribution feature, where route redistribution parameters affect both eBGP and iBGP.

You can configure an export policy to specify any of the following options:

- To reject routes that match those in the export policy (route is not exported)
- To accept routes that match those in the export policy (the route is exported without evaluating the IP Service Activator defined export policy)
- To neither reject or accept routes (the IP Service Activator defined export policy only applies)

In all cases other policy actions, such as modifying or adding route targets, will be applied to routes.

> **Note:** The user-defined export policy must be defined in the main configuration and not within the orchestream configuration group.

You configure a community to associate with exported routes using the commands described in "VPN Topology and Route Targets".

About Predefined Route List Filters

The number of routes that are received from or sent to a CE router can be selectively reduced using a manually predefined route list filter installed on the neighboring PE router. Routes whose prefixes match the condition specified by the route list filter will either be accepted or rejected by the PE router depending on the action specified by the filter. You also need to specify in the client that the route list filter is to filter routes that are either incoming (CE-PE) or outgoing (PE-CE).

Another method for selectively reducing the number of routes that are received from or sent to a CE router is to use a prefix list filter installed on the neighboring PE router. A prefix list filter exactly matches the prefixes that are listed in the prefix list, whereas a route list filter supports a range of matching conditions which can apply to specific routes or a range of routes.

Configuring a Route List Filter

The name of the preconfigured route list filter must be specified in a policy-statement. You apply the route list filter to a site by entering the name of the policy-statement in the Prefix filters In or Out fields on the Site properties, EBGP Advanced page.

To configure a route list filter on the PE device in router configuration mode:

1. Use the following syntax at the [edit routing-options policy-options] level:

```
policy-statement policy-name {
  term term-name {
    from {
      route-filter destination-prefix match-type actions;
      source-address-filter destination-prefix match-type actions;
      }
      then actions;
    }
```

```
    }
}
```

where:

- route-filter matches outgoing prefixes (PE-CE) and source-address-filter matches incoming prefixes (CE-PE)
- *destination-prefix* is an IPv4 prefix in the format of prefix/prefix length. If prefix length is not entered, default is 32.
- *match-type* is the type of match applied to *destination_prefix*. See table Table 4–1 for possible match types.
- *actions* is an action taken if destination_prefix matches. Can be one or several actions listed in Table 4–2. Actions can either be specified in the route-filter and source-address-filter sections or in the then-statement.

Table 4–1 describes the possible match types.

Table 4–1 Match Types

Type	Description
exact	The destination-prefix prefix (specified by the prefix-length) and prefix-length match the route's prefix and prefix length.
longer	The destination-prefix prefix (specified by the prefix-length) matches the route's prefix and its prefix-length is greater than the route's prefix length.
orlonger	The destination-prefix prefix (specified by the prefix-length) matches the route's prefix and its prefix-length is equal to, or greater than the route's prefix length.
prefix-length-range prefix-length-*value1* prefix-length-*value2*	The destination-prefix prefix (specified by the prefix-length) matches the route's prefix and the route's prefix length is within *value1* and *value2*.
first-destination-prefix through *last-destination-prefix*	Matches all of the following: ■ The first-destination-prefix prefix (specified by the prefix-length) matches the first route's prefix. ■ The last-destination-prefix prefix (specified by the prefix-length) matches the last route's prefix. ■ The number of bits in the first route's prefix is less than, or equal to, the number of bits in the last route's prefix.
upto prefix-length-*value2*	The destination-prefix prefix (specified by the prefix-length) matches the route's prefix and the route's prefix length is within destination-prefix prefix-length and prefix-length-*value2*

Table 4–2 describes the possible actions.

Table 4–2 Actions

Action	Description
accept	Accepts the route. After a route is accepted, no other terms in the routing policy and no other routing policies are evaluated.
reject	Rejects the route. After a route is rejected, no other terms in the routing policy and no other routing policies are evaluated.

Table 4–2 (Cont.) Actions

Action	Description
next term	Skips to and evaluates the next term in the same routing policy. Any accept or reject action specified in the then statement is skipped. This is the default action if no action is specified.
next policy	Skips to and evaluates the next routing policy. Any accept or reject action specified in the then statement is skipped.

The policy statement is referenced at the [edit routing-instances VRF-table-name protocols bgp group neighbor] level. For more information, see "Allow AS In".

Configuring a Prefix List Filter

Another method for selectively reducing the number of routes that are received from, or sent to, a CE router is to use a prefix list filter installed on the neighboring PE router. The IP address prefixes of routes that you want to filter are listed in a prefix-list. Routes whose IP address prefixes match those in the prefix-list will either be allowed or rejected by the PE router depending on the action specified in the route list filter. The route list filter has to specify that the prefix-list is required to only filter routes that are either incoming (CE - PE) or outgoing (PE - CE).

The name of the prefix-list must be specified in a policy-statement. Both the prefix-list and the policy-statement must be preconfigured on the PE device. You apply a preconfigured prefix list filter to a site by entering the name of the policy-statement in the Prefix filters In or Out fields on the Site properties, EBGP Advanced page.

To define a prefix list:

Use the following syntax at the [edit routing-options] level:

```
prefix-list name {
  ip-addresses;
}
```

You can list any number of ip-addresses.

You define a policy-statement which defines the prefix list filter and specifies the prefix-list at the [edit routing-options policy-options] level.

To define the policy-statement for outgoing (export) routes:

1. Use the following syntax, entering the previously-defined prefix list name in the from-statement. Use either **accept** or **reject** in the then actions statement to accept or reject prefixes:

```
policy-statement policy-name {
  term term-name {
    from {
      match-conditions;
      prefix-list name;
      }
    to {
      match-conditions;
        neighbor address;
    }
    then actions;
      accept;
      reject;
    }
```

```
}
```

To define the policy-statement for incoming (import) routes:

1. Use the following syntax, entering the previously-defined prefix list name in the from-statement. Use either **accept** or **reject** in the then actions statement to accept or reject prefixes:

```
policy-statement policy-name {
  term term-name {
    from {
      match-conditions;
      neighbor address;
      }
    to {
      match-conditions;
        prefix-list name;
    }
    then actions;
      accept;
      reject;
    }
}
```

A policy-statement policy-name is added at the [edit routing-instances VRF-table-name protocols bgp group neighbor] level. For more information, see "Allow AS In".

About Predefined Filtering Policies for Route Redistribution

You can specify a manually preconfigured policy statement to filter routes redistributed from the site's VRF table into BGP and routes received from other sites by BGP. A different policy can be specified for the protocol used for PE-CE connectivity, connected routes and static routes. By applying a policy to redistributed routes you can eliminate the routing loops and convergence problems that can potentially result from route redistribution.

Users apply filters to a site by specifying the name of a policy-statement in the BGP Policy and/or ProtocolName Policy field on the Site's Redist property page.

To configure a filter on the PE device in router configuration mode:

1. Use the following syntax at the [edit routing-options policy-options] level:

```
policy-statement forwarding-policy {
  term one {
    from {
      protocol protocol-name;
      route-filter destination-prefix match-type actions;
    }
      then {
        actions;
    }
  }
  term x {
    …
  }
}
```

where *destination-prefix* is the IP route prefix to match, *match-type* is the type of match (see Table 4–1), and *actions* is the action to take if destination-prefix matches (see Table 4–2).

If the policy is applied to routes distributed into iBGP, IP Service Activator applies the policy statement to the export policy for the relevant VRF table. For more information, see "Exporting Routes Between Routing Instances in Overlapping VPNs".

If the policy is applied to routes distributed into the protocol used for PE-CE connectivity, IP Service Activator applies the policy statement to the PE-CE export policy. For more information, see "Specifying Metrics for Route Redistribution".

Performing Manual Preconfiguration for MPLS Tunneling CCCs

Perform the following preconfiguration tasks for CCs on Ethernet interfaces:

- For 802.1Q VLANs and/or VLAN-based CCCs: ensure that vlan-tagging is enabled on physical interfaces and that each logical subinterface has a VLAN ID configured. However, in some JUNOS devices, it is possible for a logical subinterface to have vlan-tagging without a VLAN ID. The validation is removed from Juniper Device Driver code.
- For physical Ethernet interfaces to be used in port-based CCCs: ensure that there is no vlan-tagging and only unit 0, or none of logical subinterfaces, are present.

For more information, refer to the JUNOS Software MPLS Applications Configuration Guide, available on the Juniper Networks Technical Documentation Web site:

http://www.juniper.net/techpubs/en_US/release-independent/junos/information-products/pathway-pages/junos/product/

Configuring RSVP

Oracle recommends that you enable RSVP on all core router interfaces except for those on the Autonomous System (AS) border. RSVP must be configured manually on all routers.

To configure RSVP:

1. Use the following command syntax:

```
protocols {
  rsvp {
    interface {interface-name | all };
  }
}
```

Manually Preconfigured BGP Peering

IP Service Activator configures BGP peering in two different cases:

- PE routers in the same AS
- PE routers in more than one AS

PE Routers in the Same AS

iBGP peering must be manually preconfigured between PE routers that reside in the same AS.

For example:

```
protocols {
  bgp {
    group VPN-IBGP-Peers {
```

```
      type internal;
      local-as global-asn;
      family inet-vpn {
        unicast;
      }
      neighbor peer-loopback-address {
        local-address local-loopback-address;
      }
      neighbor peer-loopback-address {
        local-address local-loopback-address;
      }
      more neighbours...
    }
  }
}
```

PE Routers in More Than One AS

You can use IP Service Activator to manage manually preconfigured multi-AS VPNs.

When managing multi-AS VPNs with IP Service Activator, the domain-level property **Configure iBGP Peering** must be de-selected in the client. For more information, see *IP Service Activator VPN User's Guide*.

eBGP peering must be manually preconfigured between PE routers where each PE router resides in a different AS. The type is external, the update source must be explicitly set to the loopback address. The multihop option is likely required.

For example:

```
group VPN-EBGP-Peers {
  type external;
  family inet-vpn {
    unicast;
  }
  local-as 2;
  neighbor 1.1.1.1 {
    multihop;
    local-address 2.2.2.2;
    peer-as 12345;
  }
}
```

For MPLS tunneling CCCs, the nodes between the endpoints must have MPLS enabled manually. RSVP must be configured on all routers, including the LSP's ingress and egress routers. For information on enabling MPLS, see "Configuring IP and MPLS on Core Interfaces".

The device driver configures MPLS on the CCC's ingress and egress routers. MPLS must be manually configured on all intervening routers.

A loopback interface must also be configured on the LSP's ingress and egress devices to provide termination points for the CCC. For information on configuring the loopback interface, see "Configuring Loopback Interfaces".

In MPLS tunneling CCCs, the LSP is dynamic and packet routing may change if the network topology is changed. Oracle therefore recommends that you configure MPLS and RSVP on all devices that may potentially be used in the LSP.

Performing Mandatory Manual Preconfiguration for Layer 2 Martini VPNs

IP Service Activator detects incompatibilities in the physical interface encapsulation pre-existing on the router when the device driver creates a new configuration for CCCs and/or l2circuits. When this occurs, an error is flagged in the client. You are then given the option to manually correct the interface encapsulation. In order to expedite the configuration process, ensure that proper interface encapsulations are provisioned as per the JUNOS documentation and this guide.

Before using IP Service Activator to set up Layer 2 Martini VPNs, some manual configuration of routers is required. Since Martini tunnels use MPLS label switched paths, much of the necessary setup is the same as that which is required to support MPLS VPNs.

Perform the following manual configuration tasks on core routers on the path to the neighbor PE, and core interfaces:

1. Enable MPLS support on all appropriate interfaces using the following command:

```
protocols {
  mpls {
    interface {<interface-name> | all };
  }
}
```

2. Configure OSPF (or another IGP) using the following command:

```
protocols {
  ospf {
    traffic-engineering
    area <address> {
      interface <interface-id>
      interface <loopback-id>
    }
  }
}
```

3. Configure LDP. On PE devices LSPs must be configured between the loopback addresses of all PE and P devices. Use the following command:

```
protocols {
  ldp {
    interface <interface-id>
    interface <loopback-id>
  }
}
```

Performing Manual Preconfiguration for Martini Circuits on Ethernet Interfaces

For 802.1Q VLANs and/or VLAN-based l2circuits, ensure that vlan-tagging is enabled on physical interfaces and that each logical subinterface has a VLAN ID configured.

For physical Ethernet interfaces to be used in port-based Martini circuits, ensure that there is no vlan-tagging and only unit 0, or none of logical subinterfaces, are present.

For more information, refer to the JUNOS Software MPLS Applications Configuration Guide, available on the Juniper Networks Technical Documentation Web site:

http://www.juniper.net/techpubs/en_US/release-independent/junos/information-products/pathway-pages/junos/product/

5

Configuration of MPLS VPNs

This chapter describes how Oracle Communications IP Service Activator configures MPLS VPNs on Juniper M-series devices.

Prerequisites for MPLS VPN Configuration

The Juniper M-series Device Driver configures the PE routers that define the membership of a VPN. The information set up on each PE router defines the VPNs to which connected sites belong and the routes to and from these sites that are to be distributed throughout the VPN.

IP Service Activator does not configure the CE routers or the provider core routers.

Before setting up VPNs you should ensure that:

- All routers are appropriately configured. See "Preconfiguring Routers".
- Domain-level parameters are appropriately set. See "Setting Domain-level Parameters".
- All routers and their interfaces within the VPN are correctly assigned roles. See "Assigning Roles".

Preconfiguring Routers

Some preconfiguration of PE and P routers is required. For example, MPLS must be enabled. For full details of the preconfiguration required, see "Manual Preconfiguration".

Setting Domain-level Parameters

A number of BGP parameters may be set up at domain level on the VPN BGP, ASN and VPN MPLS property pages of the Domain dialog box:

- Specify whether you want IP Service Activator to set up iBGP peering on the PE devices. See "Co-existence with Previously Configured iBGP".

  The default is for IP Service Activator not to configure iBGP peering. If you leave this setting off, iBGP peering must already be configured correctly on your devices.

  If Route Reflectors are used, iBGP peering must be deselected.

  If IP Service Activator is to manage multi-AS VPNs, iBGP and eBGP peering must be configured on devices and IP Service Activator's configure iBGP peering capability must remain deselected. See "Manually Preconfigured BGP Peering".

- On the ASN property page of the Domain dialog box, set the ASN for the domain.

  If there is no ASN already configured on the device, IP Service Activator configures the device with the ASN specified in the client. If an ASN is already configured on the device, IP Service Activator ignores the ASN specified in the client and uses the one found in the configuration instead. This enables IP Service Activator to support multi-AS VPNs.

- Choose whether you will enable **Allow AS in** which allows PE devices to re-advertise route prefixes containing one or more instances of the same ASN in the AS_PATH attribute. Specify the maximum number of instances allowed for an incoming prefix to be permitted by the PE device. The PE device denies incoming prefixes having more than the number of instances specified. See "Allow AS In".

- Specify whether a global ASN applies to sites within VPNs, or whether the ASN is set at site. If a global ASN applies, you can also enable **AS Override** which allows PE devices receiving route prefixes from the core, whose AS_PATH attributes have ASNs matching the ASN of their neighboring CEs, to substitute those ASN instances with the ASN of the service provider network. Prefixes with the substituted ASNs are then re-advertised to neighboring CEs. This is enabled by default. For more information, see "AS Override".

- Choose whether you will enable load-balancing between eBGP peers by setting a value for Maximum Paths. This controls the number of alternative routes to a given prefix that are maintained in a device's routing table. By default, this option is disabled and no alternatives are held. To enable load-balancing, you specify the number of routes that are maintained. The global setting can be overridden by a site-specific value on the Site dialog box. For more information, see "eBGP Load Sharing".

- Choose whether the identity of iBGP peers and the integrity of data exchanged during iBGP sessions will be verified using MD5 Authentication. See "Configuring MD5 Authentication".

> **Note:** The **Send Communities** and the **Loopback ID** options (selected on the VPN BGP property page of the Domain dialog box) are not valid on Juniper M-series devices. If any of these values are set, they are ignored.

For more information, see *IP Service Activator VPN User's Guide*.

Assigning Roles

In an MPLS domain, the core provider network is assumed to use public addresses. All CE routers are assumed to use private addresses. An IP address or DNS name must be specified in order to discover all devices in the domain.

All devices within the network must be assigned the correct system-defined roles (that is, PE routers must be classified as Gateway devices, P routers as Core devices and CE routers as Access devices). Interfaces to be configured must also be assigned the correct roles. You can assign user-defined roles as well as the system-defined roles.

Do not use role assignment rules. You must assign a role manually for each device. For more information, see the discussions of roles of routers and policy roles in *IP Service Activator User's Guide*.

Note: Role assignment rules do not function correctly for gigabitEthernet interfaces on Juniper devices. These interfaces are discovered with the media type **ethernet-csmacd(6)** rather than **gigabitEthernet(117)**.

To work around this problem, use the media type **ethernet-csmacd(6)** when creating role assignment rules for gigabitEthernet interfaces. Note that this impacts the media type granularity of the role assignment rules.

About Routing Tables and Route Targets

Juniper devices maintain the following routing tables that relate to VPN configuration:

- **bgp.l3vpn.0** stores VPN-IPv4 unicast routes received from other PE routers through iBGP. This is where a route received from another PE router is initially placed and resolved using the inet.3 routing table.
- **routing-instance.inet.0** is the VRF table. It holds routing information for one or more access interfaces associated with a VPN. See "About VRF Tables".
- **inet.3** stores MPLS routes learned from LDP and RSVP signaling for VPN traffic. It is used by the PE device to resolve routes received from PE peers in VPN-IPv4 format. For VPN routes to be resolved properly, this table must contain routes to all the PE routers in the VPN.
- **inet.0** is the global routing table. It stores IPv4 routes learned by BGP sessions between peer devices.
- **mpls.0** is the MPLS path routing table, maintained by MPLS. It contains a list of the next label-switched router in each LSP and is used on transit routers to switch packets to the next router along an LSP.

About VRF Tables

For each interface that participates in a VPN, the PE device stores associated VPN routing information in a VRF table. The table stores:

- All unicast IPv4 routes received from directly-connected CE devices.
- All explicitly configured static routes.
- Routes learned from PE peers that match the VRF table's import policy.

Every VRF table has an import and an export policy associated with it that specifies which routes are imported into and exported from the table as follows:

- Routes learned through iBGP from other PE peers are imported into the VRF table (from the **bgp.l3vpn.0** table) if the route matches the VRF import policy.
- Routes learned from directly-connected CE devices are stored in the VRF table and exported into BGP for distribution to other PE devices if they pass the VRF export policy. On export, the PE device tags a route with one or more route targets. For information on route targets, see "VPN Topology and Route Targets".

You also have the option of applying a user-defined export map to the export policy configured by IP Service Activator. See "About Previously Defined Export Maps".

The VRF table is defined at the **edit routing-instances** level as follows:

```
vrf-table-name {
```

```
  instance-type vrf;
  interface logical-interface-name;
  route-distinguisher RD-number;
  vrf-import import-policy;
  vrf-export export-policy;
  protocols {
    protocol-configuration
  }
}
```

where *vrf-table-name* is an identifier for the VRF table, *logical-interface-name* is the name of the logical interface, *RD-number* is the route distinguisher, *import-policy* is the name of the import policy to be applied to routes received from PE peers through iBGP, *export-policy* is the name of the export policy to be applied to routes exported into BGP, and *protocol-configuration* is the protocol configuration details (RIP, OSPF, or eBGP).

> **Note:** If static routes are selected for PE-CE connectivity or if IP Service Activator has configured rib groups, an additional routing-options clause is included within the VRF definition. For more information, see "PE-CE Communication Using Static Routes" and "Exporting Routes Between Routing Instances in Overlapping VPNs".

By default, IP Service Activator generates VRF table names of the form Orch_ *RD-number*. However, you can define specific names for VRF tables on selected interfaces if you do not want to use the system-calculated ones. You must ensure that the name you enter does not match any user-defined VRF tables that may exist on the device if you want those VRF tables to be preserved. (See "Co-existence with Previously Configured iBGP").

If a user-defined VRF table name begins with a digit, IP Service Activator prepends a _ character to the name.

By default, IP Service Activator automatically generates a specific VRF table name for each interface that participates in a VPN. However, if you wish to apply the same RD number to all interfaces that participate in the VPN, the same VRF name will also apply (auto-generated or user-defined). See "Setting RD Number per VPN".

> **Note:** Do not modify VRF parameters of an unmanaged device. This is not supported by IP Service Activator.
>
> IP Service Activator may get out of sync with a device if VRF parameters are changed while a device is unmanaged, and the device is then managed. In this case, the device may not have correct VRF configuration.
>
> If this happens, manually remove the incorrect VRF configuration from the device.

About vrf-table-label Keyword

IP Service Activator supports the vrf-table-label configuration keyword on Juniper equipment. On an MPLS VPN, this feature causes the inner (VPN) label of a packet to be popped off as the packet arrives at a VRF so that it can be processed based on the contents of its IP header. Without this feature, incoming packets are mapped directly onto an outgoing (CE-facing) interface based on the inner VPN label.

Support for the Juniper vrf-table-label setting is configured in two places: for the VPN itself (on the VPN dialog box, VRF property page) and for the site (on the Site dialog box, on the VRF property page).

On the VPN dialog, VRF property page:

- To enable vrf-table-label support, check the **VRF label** option.
- To clear vrf-table-label support, clear the **VRF label** option.

On the Site dialog box, VRF property page:

- To enable vrf-table-label support, check the **VRF label** option.
- To clear vrf-table-label support, clear the **VRF label** option.

Setting Route Distinguishers

Customer networks typically use private addresses. Overlap between customer addresses may occur when they connect to the public Internet or to the provider's NOC. To avoid this problem, iBGP prefixes a site identifier, known as a route distinguisher or RD number, to each route associated with a particular site. This ensures that VPN routes are unique within the Internet.

The new route is part of the VPN-IPv4 address family: a BGP address family added as an extension to the BGP protocol.

The RD number consists of two integers separated by a colon. It can be in either of the following formats:

- *32-bit IP address*:*16-bit number*
- *16-bit ASN number*:*32-bit number*

IP Service Activator normally generates RD numbers automatically, using the ASN for the high-order number and the unique system ID of the Site object for the low-order number. For example, 1:3125

However, you can override these defaults and specify your own RD numbers if you wish.

To specify your own RD numbers:

1. At the [edit routing-instances *vrf-table-name*] level, enter the following command:

   ```
   route-distinguisher high-order;low-order
   ```

> **Note:** IP Service Activator checks that system-generated RD numbers are unique but no check for uniqueness is made on user-defined RD numbers.

The interface description of a VPN Site, entered in the Site Properties - Addressing dialog, is enabled as the interface description only if the -OverwriteInterfaceDescription parameter is present on the JDD command line. For more information on this parameter see "Command-line Parameters".

Setting RD Number per VPN

By default, IP Service Activator automatically generates a site-specific VRF table name and RD number for each site that participates in a VPN.

However, you can override the IP Service Activator default by specifying at the VPN level that the same VRF table name and RD number is applied to all sites that participate in the VPN. You can choose whether to use IP Service Activator-generated values or specify your own VRF table name and RD number.

On the IP Service Activator client, these settings are is specified on the VRF page of the VPN property dialog box.

In addition, you can override the name generation rules so that if a site is part of only one VPN, the VRF table name and RD are derived from the VPN. If a site is part of multiple VPNs, the VRF table name and RD are derived from the site.

If a single RD number or VRF table name is set per VPN, the settings for VRF re-use and reduction must also be set at VPN level. See "About VRF Re-use and Reduction".

Note: Using a single RD number for all sites in a VPN is suitable only where a site belongs to one intranet VPN. If the site may become a member of an extranet VPN in the future, Oracle does not recommend this method.

VPN Topology and Route Targets

The connectivity of the VPN can be one of the following:

- Mesh: all sites have connectivity to all other sites
- Hub and Spoke: one or more hub sites has access to all other sites; spoke sites can access the hub only
- Management: works in the same way as hub and spoke, but is used when setting up QoS to ensure connectivity to CE devices

When setting up a VPN, you have to set its connectivity, and for a hub and spoke or management VPN, select the hub site(s).

To create a fully-meshed VPN, each site's VRF table imports and exports the same routes. In a hub and spoke or management VPN the VRF table at the hub site imports routes from all other hub sites and all spoke sites, and exports routes to other hub sites and to the spoke sites. VRF tables at spoke sites export routes only to the hub site and import routes only from the hub site.

A route target (RT) identifies a set of sites within a VPN to which a PE device distributes routes.

Route targets are used to create the VPN topology. Each VPN must have a unique route target number. The RT is implemented as a BGP extended community. A BGP community groups a set of destinations that share a common property. In this case the property is a set of routes that are to be distributed to a set of CE sites. The RT is added to the route by the ingress PE device and used by the egress PE device to determine whether a received route is destined for a VPN that the PE services.

IP Service Activator creates one or more BGP communities per VPN, depending on the VPN topology:

- If the VPN is fully-meshed, IP Service Activator creates one community. Every site receives routing information from all other sites.
- If the VPN is a hub and spoke or management VPN, IP Service Activator creates two communities. There are effectively two sets of devices: one set of hub site(s) and one set of spoke sites. By default, routes from the spoke sites are only

distributed to the hub site(s), routes from the hub site(s) are distributed to all spoke sites and any other hub sites.

The RT number consists of two integers separated by a colon. It can be in one of two alternative formats:

- *32-bit IP address*:*16-bit number*
- *16-bit ASN number*:*32-bit number*

IP Service Activator normally generates RT numbers automatically, using the ASN for the high order number and the unique system ID of the VPN for the low order number. For example, 20:4926.

In a hub and spoke VPN topology, IP Service Activator generates two RT numbers: one for the hub site(s), generated as indicated above, and one for all spoke sites, generated by incrementing the hub low order number by one.

If you wish, you can specify your own RT numbers for hub, spoke, or fully-meshed sites within a VPN if you do not want to use the system-generated default values. You can easily reassign RT numbers to sites within a VPN, if for example, it has been imported from a different system or it is to be exported to a different system.

You can specify any number of RT values per VPN and specify whether a value is used for VRF import, VRF export, or neither, for hub, spoke and fully-meshed behaviors. On the client, these settings are on the MPLS property page of the VPN dialog box.

Note: IP Service Activator checks that system-generated RT numbers are unique but no check for uniqueness is made on user-defined RT numbers.

The community is configured at the [edit groups orchestream policy-options] level as follows:

```
community community-name members target:route-target-value;
```

where community-name is an identifier for the community and route-target-value identifies the route target in the format *AS-no* or IPAddress:*ID*. *AS-no* is the domain AS number and *ID* is an identifier based on the site's object ID number within IP Service Activator.

By default, in a hub and spoke VPN, hub sites are members of community _0 and _1, and spoke sites are members of community _1, as follows:

```
community VPN-name_0 members target:route-target-value
community VPN-name_1 members target:route-target-value
```

For each new user-defined route target, IP Service Activator allocates a community value that is incremented by one, for example, if another route target is added to a VPN with the above route targets, it will be allocated to community _2.

A community is referenced within the policy statement for each VRF table's import and export policy.

About VRF Import and Export Policies

The VRF import and export policies define which routes are imported into and exported from the VRF table.

For information on the PE-CE export policy, see "Route Redistribution".

Configuring the Export Policy

The VRF export policy specifies whether a route received from a directly-connected CE device is advertised by the PE device to other peer PE devices. There is one export policy per VRF table.

Routes received by the PE device from a directly-connected CE device are placed in the VRF table. The device checks the received route against the VRF export policy for that interface. The policy consists of three terms:

- The first term specifies the community to be added to the route on export.
- The second term specifies from which protocol routes must be received for the policy to be applied.
- If the VRF table belongs to an interface whose VPN membership overlaps with that of another interface on the PE device, an extra interface clause is defined in the term. For more information, see "Exporting Routes Between Routing Instances in Overlapping VPNs".
- The final term rejects any routes that do not meet the protocol criteria.

If a route passes the VRF export policy criteria:

- A route target is added to the route – several route targets may be added if an interface is associated with multiple VPNs.
- The route is converted to VPN-IPv4 format.
- The route is advertised through iBGP to peer PE devices within the VPN.

By default, the VRF export policy for hub sites specifies community_0, and for spoke sites, community_1.

IP Service Activator names the VRF export policy in the following format:

`Orch_`*`RD-number`*`-export-to-PEs`

where *RD-number* is the route distinguisher associated with a site.

If users have specified metrics for route redistribution from the PE to CE protocol into BGP, a policy statement is configured and referenced in the VRF export policy. For more information, see "Exporting Routes Between Routing Instances in Overlapping VPNs". It is also possible to manually pre-configure a policy statement to apply to redistributed routes. For more information, see "About Predefined Filtering Policies for Route Redistribution".

The VRF export policy is applied to a VRF by the vrf-export statement. For information, see "About VRF Tables".

The VRF export policy is configured at the [edit groups orchestream policy-options] level as follows:

```
policy-statement vrf-export-policy-name {
  term add_communities {
    then {
      community add export-community-name;
    }
  }
  term export {
    from {
      protocol [ protocol-list ];
    }
    then accept;
  }
```

```
  term reject-others {
    then reject;
  }
}
```

where *vrf-export-policy* is the identifier for the VRF export policy definition (Orch_*RD-number*-export-to-PEs), *export-community-name* is the community name (route target) to be added to the exported routes, and *protocol-list* is the list of protocols used for PE to CE connectivity within a VPN.

If an interface is associated with multiple VPNs, a community add statement is defined for each VPN. For example:

```
term add_communities {
  then {
    community add France;
    community add Europe;
  }
}
```

where France and Europe are VPN names.

Configuring the Import Policy

The import policy defines whether a route advertised by a peer PE device is imported into a VRF table. There is one VRF import policy per VRF table.

The policy consists of two terms:

- The first term accepts routes received via iBGP from other PE devices that are tagged with the specified route targets
- The second term rejects any routes that are not accepted by the first term

If a route passes the import criteria, it is added to the VRF table.

By default, the VRF import policy for hub sites specifies community_0 community_1, and for spoke sites, community_1.

IP Service Activator names the VRF import policy in the following format:

Orch_*RD-number*-import-from-PEs

where *RD-number* is the route distinguisher associated with a site. For information about route distinguishers, see "Exporting Routes Between Routing Instances in Overlapping VPNs".

The VRF import policy is referenced at the [edit routing-instances vrf-table-name protocols] level and configured at the [edit groups orchestream policy-options] level as follows:

```
policy-statement vrf-import-policy-name {
  term import {
    from {
      protocol bgp;
      community [ community-list ];
    }
    then accept;
  }
  term reject-others {
    then reject;
  }
}
```

where *vrf-import-policy-name* is the identifier for the VRF import policy definition (Orch_*RD-no*-import-from-PEs) and *community-list* is a list of one or more communities, depending on how many VPNs the interface is associated with.

Exporting Routes Between Routing Instances in Overlapping VPNs

Starting with IP Service Activator version 4.2, the JUNOS auto-export command is used by default to implement policy-based export of routes between routing instances in overlapping MPLS VPNs. This greatly simplifies the device configuration required to set up overlapping VPNs.

If you want to override the use of the auto-export command and continue to use RIB groups, use the -UseRIBGroup command line parameter. See "Exporting Routes Between Routing Instances in Overlapping MPLS VPNs".

The auto-export command comes into play when:

- Two or more PE interfaces are associated with some, but not all, of the same VPNs.
- Two or more PE interfaces are associated with the same VPNs, but at least one interface has been given ownership of its VRF table to prevent VRF re-use optimization being performed by IP Service Activator. For more information, see "About VRF Tables" and "About VRF Re-use and Reduction".

In these instances, routing information is not exchanged by iBGP, as the exchange is between VRF tables located on the same PE device. Auto-export enables routing information to be exchanged between VRF tables that are located on the same PE device.

For more information, see "PE-CE Communication Using eBGP".

About VRF Re-use and Reduction

A VRF table is set up on the device for each PE interface that is a member of a VPN. However, by default, if multiple VRF tables contain exactly the same routes (for example, if one site connects to two interfaces, or there are two sites that are members of the same VPN) IP Service Activator normally reduces them to just one, in order to minimize resource usage. This is known as VRF re-use.

In some cases automatic VRF re-use may not be required. For example, you may want to provision dual links to customer sites in order to implement load balancing, requiring a separate VRF table for each connecting interface, or to reduce the impact of future reconfiguration. In this case you can override VRF re-use by specifying that particular interfaces are always to have their own VRF table, and by specifying that other VRF tables are allowed to be merged with this VRF table by selecting the Shareable option.

On the client, you can select the **Force Install** and **Shareable** options per interface on the VRF property page of the Site dialog box.

If you are setting up a single VRF table name or RD number per VPN, settings for VRF re-use are made at the VPN level. In this case it is not possible to select **Force Install**. See "Setting RD Number per VPN".

Note: Specifying a user-defined VRF table name affects how IP Service Activator performs VRF reduction. Where system-defined VRF table names are used, VRF reduction is based on the site's RD number and a site with a lower RD number takes precedence over a site with a higher RD number. Where user-defined VRF table names are used, IP Service Activator performs VRF reduction based on table names

For complete details on VRF re-use and reduction, see *IP Service Activator VPN User's Guide*.

About Previously Defined Export Maps

The **Export map name** option allows you to specify the name of a manually pre-configured export policy to be applied to routes exported to PE peers. The policy is associated with the VRF table for the selected interface. The user-defined export policy is referenced in the VRF table's export policy. This allows greater flexibility for determining which routes are exported. You can assign the same user-defined export policy to several VRFs.

For information on setting up an export map, see "About Predefined Export Maps".

About Load Balancing in Layer 3 VPNs

IP Service Activator supports the multipath statement for the Juniper Device Driver. The **Multipath**, **VpnUnequalCost**, and **EqualExternalInternal** fields support load balancing in Layer 3 VPNs. This feature is supported only on Juniper Device Driver.

Configuring PE-PE Peering

In order to exchange information throughout the VPN, each PE router needs to run an iBGP session with each other PE router connected to a site within the same VPN.

You must configure this manually.

Configuring iBGP

iBGP is the protocol used for communication of VPN routes between PE devices in an MPLS VPN. In order for devices to exchange routing information, an iBGP session must be configured between the PE devices that comprise the VPN.

Co-existence with Previously Configured iBGP

iBGP peering should be manually configured. When the **Configure IBGP Peering** option at the domain level is deselected on the IP Service Activator client the configuration will not be touched.

Note: Do not use IP Service Activator to configure iBGP.

Configuring MD5 Authentication

The identity of iBGP peers and the integrity of data exchanged during iBGP sessions is verified using MD5 authentication, a digital signature algorithm. Users specify a key of up to 255 characters to generate a checksum of the iBGP data that is to be sent from

a PE device to its peer. The iBGP data and its checksum are then sent to the peer device using TCP. The recipient device uses MD5 and the same key used by the sender to generate a checksum of the received iBGP data. If both checksums match, the identity of the sender and the integrity of the received iBGP data is verified.

On the IP Service Activator client, this is controlled by the domain-level parameter MD5 Authentication. If this option is selected, IP Service Activator configures an MD5 authentication key on each iBGP peer.

IP Service Activator configures the MD5 authentication key at the [edit protocols bgp group group-name neighbor address] level as follows:

```
authentication-key key;
```

where *key* is the authentication password. The password can be up to 255 characters and can comprise any ASCII strings.

Configuring MPLS on Interfaces (Access Only)

IP Service Activator configures MPLS on logical interfaces associated with a VPN.

MPLS is enabled on an interface at the [edit interfaces] level and the [edit protocols] level:

The following configuration is added to the PE device:

```
interfaces {
  vpn-access-interface {
    unit unit-number {
      family mpls;
    }
  }
}
protocols {
  mpls {
    interface logical-interface-name;
  }
}
```

where *vpn-access-interface* is the name of the logical interface's parent interface, *unit-number* is the number of the logical interface, and *logical-interface-name* is the name of the logical interface.

Interface IP Address

If the IP address specified for an interface through IP Service Activator differs from the currently-configured values on the device, IP Service Activator changes the address within the main configuration. Note that the change is commented:

```
interfaces {
  vpn-access-interface {
    unit unit-number {
      family inet {
        /* Orchestream has changed this address */
        address public-or-private-ip-address;
      }
    }
  }
}
```

This change is made in the main configuration because any interface addresses defined in the orchestream group are merged with those in the main configuration.

About PE-CE Communications

This section describes the different ways in which PE-CE communication can be configured.

PE-CE Communication Using eBGP

In order to exchange information to and from customer sites in the VPN, each PE router also needs to communicate with each of its external neighbors (the CE routers to which it is connected).

The effect is to advertise network reachability information between the CE and the PE, which in turn converts IPv4 addresses to VPN-IPv4 addresses for traffic passing from the CE to the PE and vice versa.

The details here explain the configuration of the PE routers using eBGP. The corresponding configuration of the CE routers is not performed by IP Service Activator. See "Manually Configuring CE Routers".

As well as eBGP, IP Service Activator supports RIP and OSPF routing protocols. Static routes can also be defined (either alone or in conjunction with another protocol).

If you use eBGP, specify:

- The ASN of the site
- The IP address of the corresponding interface on the CE router

You can optionally specify:

- The number of times the same ASN can appear in an incoming prefix for it to be accepted by the site PE or all PEs in the domain
- AS Override for the site
- Authentication for a PE-CE session

Basic Juniper Commands

eBGP is configured at the [edit routing-instances vrf-table-name protocols] level as follows:

```
bgp {
  group EBGP-to-CEs {
    type external;
    export CE-export-policy;
    local-as local-AS-number;
    neighbor CE-address {
      description "EBGP to site: site-name";
      local-address VPN-interface-address;
      peer-as AS-no;
  }
}
```

where *CE-export-policy* is the name of the export policy to be applied to routes being exported into BGP, *local-AS-number* is the local AS number, *CE-address* is the IP address of a CE peer, *site-name* is the name of the site as defined in the client, *VPN-interface-address* is the address of the access interface on the gateway device that is used to accept the incoming connections to the PE device and to establish connections to the CE device, and *AS-no* is the AS number of the site.

The following sections describe options which can be defined within the **neighbor** definition.

eBGP-to-CEs

If eBGP is selected as the routing protocol, IP Service Activator creates a BGP group named EBGP-to-CEs which contains:

- The type external directive – specifies that all peers are in an external BGP group, that is, they are in different Autonomous Systems
- The PE-CE export policy
- The local AS number
- A neighbor statement for each of the CE devices with which the PE device communicates. Each neighbor statement specifies:
  - The address of the local end of the BGP session
  - The peer system's AS number

Allow AS In

Some topologies require the core network AS number to occur several times in the AS_PATH attribute of a route prefix advertised to the PE device. For example, if two CE devices are connected to each other by BGP, and each belongs to a different VPN, routes advertised by the PE to one of the CEs are also advertised to its CE neighbor who advertises the routes back to the core network. This means that the PE receives routes that have its AS number. Normally, this indicates that a routing loop has occurred, and the PE denies these routes. However, to allow VPN configurations to work, the PEs have to accept these routes.

The maximum number of times the same ASN is allowed in an incoming route for it to be accepted by the PE device is enabled using the loops option when configuring the ASN at the [edit routing-instances *vrf-table-name* routing-options] level as follows:

```
autonomous-system AS-no loops loops-no;
```

The value of *loops-no* can be from 1 to 10 inclusive; the default is 0 (unconfigured).

The loops option applies to the entire VRF table and therefore to all the neighbor PE devices that the VRF connects to. This may cause undesirable behavior if VRF tables using iBGP which have different loop values are allowed to merge. A condition has been added to IP Service Activator's VRF table reduction logic to prevent this.

Because of a JUNOS restriction, when setting the AS number, the same value must be used for the loops parameter of all sites in the VPN rather than independently per BGP neighbor. Ensure that the same value is set in IP Service Activator, or set values globally per device outside IP Service Activator. On the client, this is set on the Allow AS in: Use parameter on the EBGP page of the Site dialog box.

Because of a JUNOS restriction, only the values 0 and 1 are accepted for Allow AS in. On the Juniper device, Allow AS in configures the autonomous-system loop attribute at the routing-instance routing-options CLI level.

AS Override

You can specify that the ASN of a provider is used to override the ASN of a site. When AS override is turned on, a PE device that receives route prefixes whose AS_PATH attributes have one or more ASNs matching the ASN of its neighboring CE devices, substitute those ASN instances with the ASN of the service provider network. Prefixes with the substituted ASNs are then re-advertised to neighboring CE devices. The PE device also adds its ASN to routes before exporting them to the CE device.

This allows CE devices to accept routes that have been re-advertised by devices having the same ASN, and which would otherwise be rejected. Normally, a CE device rejects routes whose AS_PATH attribute contains ASNs matching its own ASN, to prevent routing loops.

Within IP Service Activator, AS override can be specified at domain level, to apply to all sites, or set up for individual sites.

If this option is selected, the following command is configured:

```
as-override;
```

Local Preference

Where multiple PE interfaces are associated with a site, the local preference for an interface can be set. The preference value may be between 1-4294967295, and the higher the value the higher the priority. The default is Router Default (100). Local preference is configured by means of a route map, which can include other conditions.

In the IP Service Activator client, the local preferences parameter is set on the EBGP page of the Site dialog box.

If this option is selected, the following command is configured:

```
local-preference value;
```

where *value* is in the range 0 to 4 294 967 295.

Authentication

The identity of eBGP peers and the integrity of data exchanged during eBGP sessions can be verified using Authentication. If this option is selected, the following command is configured:

```
authentication-key "private-key";
```

Route Prefix Filters

You can configure a route list filter which filters incoming (CE-PE) and or outgoing (PE-CE) routes. The route list filter must be manually configured on the device as a policy-statement. Alternatively, you can specify an import or an export policy-statement which specifies a prefix list for filtering either incoming (CE-PE) or outgoing (PE-CE) routes. The prefix list must be manually configured on the device. The policy-statement and the specified prefix list must both exist on the device before the policy-statement can be implemented by IP Service Activator.

You apply a route list filter or a prefix list filter to a site by entering the name of its policy-statement in the **Prefix filters In** or **Out** fields on the Site properties, EBGP Advanced page:

```
import policy-statement policy-name;
export policy-statement policy-name;
```

where *policy-name* is the policy-statement defining the routing policy using a predefined prefix-list.

If you add the policy statement to the device after it is provisioned by IP Service Activator, you must either manually add the appropriate import and/or export policy statement(s) to the relevant group in the IP Service Activator configuration, or allow IP Service Activator to re-propagate the configuration to the device.

For information about configuring a route list filter, see "Configuring a Route List Filter". For more information about configuring a prefix list filter, see "Configuring a

Prefix List Filter".

Prefix Limit

You can specify the maximum number of eBGP IP address prefixes that the PE device is allowed to receive from a CE device. You can also allow a warning message to be generated in the device log either when the prefix limit is reached or when a percentage of the prefix limit value is reached. The maximum value you can specify is 4 294 967 296:

```
family inet {
  unicast {
    prefix-limit {
      maximum prefix-limit-value;
      teardown percentage-value;
```

where *prefix-limit-value* specifies the maximum allowed number of prefixes and *percentage-value* specifies that a device log warning message is generated at a percentage of *prefix-limit-value*.

If a warning only is specified, the **teardown** *percentage-value* line does not appear

Site of Origin

If a site is multi-homed, that is, there are multiple links to a site, it is possible for routing loops (when routes learned from a site are advertised back to that site) to occur.

To prevent routing loops, when using eBGP, IP Service Activator automatically configures Site of Origin (SOO) as follows:

- An additional BGP extended community is applied to routes learned from the CE device and imported into the relevant interface's VRF table.
- An additional import policy is defined to tag routes with the BGP extended community.
- An additional export policy is defined that rejects routes tagged with the BGP extended community. This prevents the route being advertised back to the CE device from which it originated.
- AS override is configured by default so that the site's AS number is substituted with the AS number of the service provider. This ensures that routes are not rejected by sites that share the same AS number. However, AS override can be disabled if required. For more information, see "AS Override".

The Site of Origin is defined as follows:

```
origin:SOO;
```

where *SOO* is an extended community comprising the site's ASN and internal object ID number.

IP Service Activator uses the following naming conventions:

- For the Import policy: Orch-site-name-add-SOO
- For the Export policy: Orch-site-name-deny-SOO

The following configuration is added at the [edit groups orchestream policy-options] level:

```
policy-statement SOO-import-policy {
  then {
```

```
    community add SOO-community-name;
  }
policy-statement SOO-export-policy {
  from community SOO-community-name;
  then reject;
}
    community SOO-community-name members SOO-community-value;
}
```

where *SOO-import-policy* is the name of the routing policy that is to be applied to routes being imported into the interface's VRF routing table from BGP, *SOO-community-name* is the community name (Orch-*Site-name*-SOO), *SOO-export-policy* is the name of the routing policy that is to be applied to routes being exported from the interface's RF routing table, and *SOO-community-value* is the community definition.

The import and export policies are referenced within the eBGP protocol configuration, at the [edit routing-instances vrf-table-name protocols bgp group neighbor] level:

```
import SOO-import-policy;
export [ SOO-export-policy CE-export-policy ];
```

The CE export policy is repeated at this level as the export statement defined at the neighbor level overrides the group level export statement.

AS override is configured at the [edit routing-instances <vrf-table-name> protocols bgp group neighbor] level for relevant sites:

```
as-override;
```

eBGP Load Sharing

You can enable load-balancing between eBGP peers. This allows BGP to select more than one eBGP path to a given prefix over which traffic can be shared. Routes for each path are maintained in a PE device's routing table. By default, this option is disabled and no alternative routes are held.

In the IP Service Activator client, the eBGP maximum paths parameter is set at global level on the VPN BGP page of the Domain dialog box and overridden for specific devices by a setting on the EBGP Adv. page of the Site dialog box.

JUNOS does not apply the Maximum Paths value, it only enables multiple paths to be configured and maintained if a value greater than 1 is specified.

Load-balancing is configured at the [edit routing-instances *routing-instance-name* protocols bgp group *group-name*] hierarchy level by the statement:

```
multipath;
```

Route Dampening

Route dampening is a mechanism that attempts to minimize network instability by suppressing the advertisement of poorly-behaved routes. Penalties are applied when a route is withdrawn, readvertised or changed. When a predefined penalty limit is reached, further advertisement of the route is suppressed. The penalty is reduced according to a defined half-life setting, and once the penalty decreases below a limit, the route can be readvertised.

On the IP Service Activator client, route dampening is set up on the EBGP Damp. property page of the Site dialog box.

If this option is selected, the following commands are configured at the [edit policy-options] level:

```
damping dampingName {
  disable;
  half-life minutes;
  reuse number;
  suppress number;
  max-suppress minutes;
}
```

were *dampingName* is the identifier for the group of damping parameters (Orch_*RD-no*), half-life *minutes* is the time, from 1 to 45 minutes, at which a penalty applying to a route is decreased by half, **max-suppress** *minutes* is the maximum time, from 1 to 720 minutes, that a route can be suppressed.

When a penalty applying a route falls below **reuse** *number*, which can be from 1 to 20000, the route becomes unsuppressed. When a penalty applying to a route exceeds **suppress** *number*, which can be from 1 to 20000, the route becomes suppressed.

Damping is applied by a policy statement which is configured at the [edit policy-options] level as follows:

```
policy-statement PolicyName {
  then damping DampingName;
}
```

where *PolicyName* is the identifier for the group of damping parameters (Orch_*RD-no*-EBGP-damper) and *DampingName* is the identifier for the group of damping parameters.

The policy statement may be applied to all eBGP peers or to a specific eBGP neighbor, depending on the parameters configured in the client. Damping is applied to all eBGP peers at the [edit routing-instances *vrf-table-name* protocols bgp group] level as follows:

```
import PolicyName;
```

Damping is applied to a specific eBGP neighbor at the [edit routing-instances *vrf-table-name* protocols bgp group neighbor] level as follows:

```
import PolicyName;
```

PE-CE Communication Using RIP

If RIP is selected as the routing protocol, IP Service Activator creates a RIP group named RIP-to-CEs which contains:

- An export policy statement that applies to all members of the group
- A neighbor statement for each of the interfaces over which RIP is running

Where VRF re-use has been carried out, a number of neighbor statements are defined.

IP Service Activator adds the following configuration at the [edit groups orchestream routing-instances *vrf-table-name* protocols] level:

```
rip {
  group RIP-to-CEs {
    export CE-export-policy;
    neighbor VPN-interface;
  }
}
```

where *CE-export-policy* is the name of the export policy to apply to routes being exported from the routing table into BGP and *VPN-interface* is the name of the logical interface over which the PE device communicates with a CE neighbor.

> **Note:** RIP does not support routing table groups. This means that if a CE device belongs to more than one VPN, the VRF tables on the neighboring PE device cannot share the same routes.

PE-CE Communication Using OSPF

If OSPF is selected as the routing protocol, IP Service Activator adds the interface to the OSPF backbone, area 0.0.0.0.

Routes advertised from other sites in the VPN that are not using OSPF for PE-CE connectivity are advertised as AS-External routes. BGP routes distributed into OSPF carry a default metric of 20. OSPF area numbers and Link State Advertisement (LSA) types are carried over the MPLS backbone using the BGP extended community attribute 0x8000.

> **Note:** IP Service Activator does not support use of OSPF as the PE-CE routing protocol in non-broadcast multi-access networks

IP Service Activator configures the following:

- An export policy statement that applies to the area
- An interface statement for each of the interfaces over which OSPF is running

Where VRF re-use has been carried out, a number of interface statements are defined.

IP Service Activator adds the following configuration at the [edit groups orchestream routing-instances *vrf-table-name*] level:

```
protocols {
  ospf {
    area 0.0.0.0 {
      export CE-export-policy;
      interface VPN-interface;
      …
    }
  }
}
```

where *CE-export-policy* is the name of the export policy to apply to routes being exported from the routing table into OSPF and *VPN-interface* is the name of the logical interface over which the PE device communicates with a CE neighbor.

OSPF Domain Tag

Using OSPF as the routing protocol between a CE and PE device in an MPLS VPN can result in routing loops if OSPF routes can be passed between PEs in the same network and VPN.

IP Service Activator supports VPN Route tags when OSPF is selected as a routing protocol. The VPN Route tag is one mechanism that can be used to prevent routing loops in multi-homed VPNs.

The **Domain Tag** field appears on the Site Properties dialog, Connectivity page.

When the Domain Tag field is selected, a tag value must be specified. This causes a VPN Route tag (domain-vpn-tag in Juniper devices) to be configured. In this case, the PE includes the tag in all LSA-5 advertisements sent to attached CE devices. The tag value must be distinct from any OSPF VPN Route Tag being used in the OSPF domain.

Using a different VPN Route tag causes the advertisement not to be distributed through the OSPF area to another connected domain.

PE-CE Communication Using Static Routes

If static routing is selected, IP Service Activator configures a static route between the PE and CE devices. The following statement is added at the [edit groups orchestream routing-instances *vrf-table-name*] level:

```
routing-options {
  static {
    route destination-prefix/prefix-length next-hop address;
  }
}
```

where *destination-prefix* is the network portion of the destination IP address, *prefix-length* is the destination prefix length/subnet mask, and *address* is the IP address or interface name for the next-hop router.

> **Note:** Routing table groups are not supported by static routing on some versions of JUNOS. Where VRF tables with overlapping VPN membership use static routes, IP Service Activator configures additional static routes in place of those that would have been exchanged by routing table groups.

Specifying the Location of the Next-hop Address

You can specify that the next-hop address is in the default routing table and not the VRF table by selecting the **Global** option on a site's Static Routing page. IP Service Activator adds this information to the static route at the [edit routing-instances *VRF-name* routing-options static route] level as follows:

```
next-table inet.0;
```

Specifying that a Route is Permanent

You can specify that a static route is maintained in the VRF table when the protocol process shuts down by selecting the **Permanent** option on a site's Static Routing page.

IP Service Activator adds this information to the static route at the [edit routing-instances *VRF-name* routing-options static route] level.

```
retain;
```

Route Redistribution

By default, most dynamic routing protocols only distribute routes that have been learned via the same protocol. The PE-CE export policy ensures the exchange of routing information between CE devices and PE peers within the same VPN no matter what routing protocol or method is used for PE-CE connectivity.

Note: If you configure a VPN which includes sites whose interfaces are on devices running different versions of JUNOS, there can be redistribution issues. This incompatibility occurs between Junos 6.0 and releases earlier than Junos 5.71

The following routes can be distributed to all CE devices within the same VPN:

- Routes received via iBGP from other PE devices
- Routes from all interfaces in the VRF
- Routes learnt via rib groups from interfaces in other VRF tables
- Static routes
- Connected routes

The PE-CE export policy includes one or more terms, depending on the options selected in the client. A term is always configured for BGP.

IP Service Activator applies the following default metrics to redistributed routes:

- EBGP:
  - A metric of 0 is applied to connected and static routes distributed into BGP
- RIP:
  - A metric of 0 is applied to connected and static routes and RIP routes distributed into BGP
  - A metric of 0 is applied to connected and static routes distributed into RIP
  - A metric of 1 is applied to BGP routes distributed into RIP
- OSPF:
  - A metric of 0 is applied to connected and static routes and OSPF routes distributed into BGP
  - A metric of 20 is applied to connected and static routes and BGP routes distributed into OSPF

You can specify the metric to associate with routes distributed from the selected PE-CE routing protocol into other Internal Gateway Protocols (IGPs) and BGP, and vice versa and associate a manually pre-configured policy statement to routes to filter their redistribution.

The PE-CE export policy is configured at the [edit policy-options] level as follows:

```
policy-statement PE-CE-export-policy {
  …
}
```

where *PE-CE-export-policy* is the identifier for the PE-CE export policy definition in the form Orch_*RD-no*-redistribute-*protocol*, where *RD-no* is the route distinguisher of the site and *protocol* is the name of the protocol used for PE-CE connectivity.

The PE-CE export policy consists of one or more **term** statements that define the protocols from which routes must be redistributed into the protocol used for PE-CE connectivity. The format of a statement depends on whether a metric applies to the redistributed routes.

If a metric applies to routes redistributed from a protocol, the term statement is configured as follows:

```
term export-protocol {
  from protocol protocol;
  then {
    metric value;
    accept;
  }
}
```

where *protocol* is the name of the source protocol from which routes are redistributed and *value* is a metric to be applied when redistributing routes into the PE-CE protocol.

If a metric does not apply to routes redistributed from a protocol, the term statement is configured as follows:

```
term export {
  from protocol [export-list];
}
```

where *export-list* is the list of source protocols from which routes are redistributed. The list contains only those protocols with which no metric is associated.

The CE-PE export policy always contains a term statement that defines how routes that do not match any other term are treated:

```
term reject-others {
then reject;
}
```

Specifying Metrics for Route Redistribution

You can specify the metric to associate with routes distributed from BGP into the routing protocol used for PE-CE connectivity. You can also filter and refine the redistribution of routes by associating a manually pre-configured policy statement with redistributed routes.

The client allows you to define parameters for distribution of routes from all protocols potentially used by sites within the VPN into the PE-CE protocol. In general, however, only the value entered for BGP affects device configuration. A value specified for redistribution from any other protocol affects configuration only where two or more interfaces on the PE device participate in the same VPN, use different protocols for PE-CE connectivity and share the same VRF table.

If a metric has been defined for routes redistributed into the protocol used for PE-CE connectivity, the term statement at the [edit policy-options policy-statement *PE-CE-export-policy*] level is configured as follows:

```
term export-protocol {
  from protocol protocol;
  then {
    metric <value>;
    accept;
}
}
```

where *protocol* is the name of the source protocol from which routes are redistributed and *value* is a metric to be applied when redistributing routes into the PE-CE protocol.

If a manually preconfigured policy statement has been associated with the routes redistributed from a protocol, the term statement contains an extract from clause as follows:

```
term export-protocol {
  from {
    protocol protocol;
    policy PolicyStatement;
  }
…
}
```

where *PolicyStatement* is the name of the manually preconfigured policy statement to associate with redistributed routes.

6

Configuring Layer 2 Martini VPNs

This chapter describes how Oracle Communications IP Service Activator configures Layer 2 Martini VPNs on Juniper M-series devices.

Refer to *IP Service Activator VPN User's Guide* for a technical description of Layer 2 Martini VPNs.

Overview of Layer 2 Martini VPNs

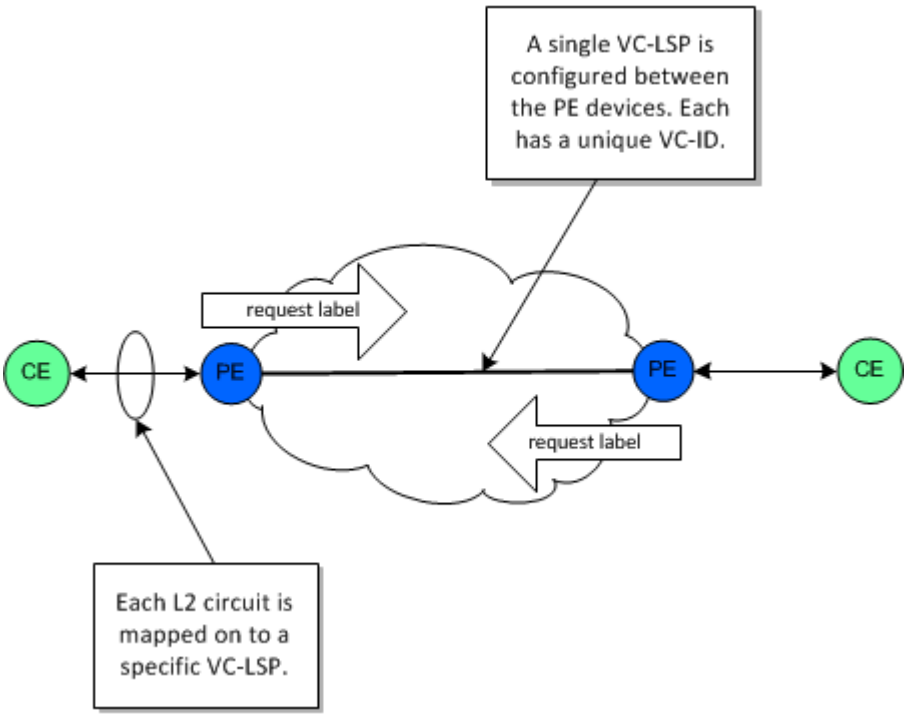
IP Service Activator supports the configuration and management of Layer 2 Martini VPNs.

A Layer 2 Martini point-to-point connection is a pseudo-wire (or tunnel) configured between two endpoints across an IP network.

The connection uses MPLS labels to encapsulate and transport various Layer 2 data formats, including Ethernet (Port), Ethernet (VLAN), Frame Relay, ATM Cell and ATM AAL5, across an IP network. The tunnel provides a transparent connection, so users see no change in their Layer 2 data. (Note that the tunnel does not aim to meet QoS aspects of the connection, particularly in the ATM case.) The Martini endpoints can be interfaces, sub-interfaces, or other endpoint identifiers (VCI/VPI on ATM, DLCI on Frame Relay, or VLAN ID on Ethernet).

A Layer 2 Martini VPN is an association of Layer 2 Martini point-to-point connections, as illustrated in Figure 6–1.

Figure 6–1 Martini Point-to-Point Links



The data types listed in Table 6–1 can be encapsulated on Layer 2 Martini VPNs on Juniper devices.

Table 6–1 Layer 2 Martini VPN Data Types

Encapsulated Data	Endpoint	Comment
Ethernet (port-based)	Main interface	NA
Ethernet (VLAN-based)	Sub-interface with VC identifier	Created by Provision sub-interface
ATM Cell	Sub-interface with VC identifier	Created by Provision sub-interface
ATM AAL5	Sub-interface with VC identifier	Created by Provision sub-interface
Frame Relay	Sub-interface with VC identifier	Created by Provision sub-interface

All Layer 2 endpoints (such as DLCI, VLANs, VPI/VCIs) and their parents (logical and physical interfaces) must have the **Access** role assigned.

ATM Cell Relay Layer 2 Martini tunnel endpoints must have the same VPI/VCI. ATM AAL5 tunnel endpoints are not required to have the same VPI/VCI.

When creating a Layer 2 Martini VPN with SONET interfaces on Juniper M-series devices as endpoints, the MTU values must match. Note that this must be set manually. IP Service Activator does not validate the MTU values, so you will not be notified when there is a potential mismatch. The Martini circuit will be created in the client but may not be operational if the MTU values do not match on the SONET endpoints.

About Encapsulation Protocols

Layer 2 Martini VPNs use MPLS, particularly to create the LSPs over which encapsulated data travels. The encapsulation protocols are described in various IETF

drafts including draft-martini-l2circuit-encap-mpls, draft-martini-l2circuit-trans-mpls, draft-martini-ethernet-encap-mpls, and draft-martini-atm-encap-mpls. See "Martini Drafts" for details.

Base Configuration

Some manual preconfiguration of devices is required to support the configuration of Layer 2 Martini VPNs in IP Service Activator.

Core routers on the path to the neighbor PE, and core interfaces, must run MPLS, an IGP such as OSPF, and LDP.

Setting Up Layer 2 Martini VPNs

Complete the following tasks in order to set up a Layer 2 Martini connection:

you perform preconfiguration tasks, create the endpoints (either sub-interfaces or VC interfaces), create the Layer 2 Martini tunnel object, set the appropriate options, and finally, assign the relevant endpoints to the tunnel.

1. Set up base configuration: before actually provisioning the Layer 2 Martini tunnel, certain preconfiguration tasks must be performed. See "Base Configuration".
2. Create the endpoints: endpoints are interfaces or sub-interfaces supporting the required type of data encapsulation. See "Provisioning Sub-interfaces for a Layer 2 Martini Connection".
3. Create the VPN: add the Layer 2 Martini connection by right-clicking on the Point-to-Points for the customer and selecting **Add L2 Martini-Pt-Pt**. Complete the specifications in the dialog box, and then add the endpoints to the new Layer 2 Martini tunnel.
4. Check the configuration: use Telnet to directly access the running configuration on the devices. See "Checking the Configuration Added to the Device".

Provisioning and Deleting Sub-interfaces

This section explains how to provision and delete Layer 2 Martini sub-interfaces.

Provisioning Sub-interfaces for a Layer 2 Martini Connection

Layer 2 Martini VPNs provide a tunnel between two endpoints which carries encapsulated data. The encapsulation is done at the endpoints, or sub-interfaces. The sub-interfaces provisioned to support the Layer 2 Martini tunnel must support the desired type of data encapsulation.

See *IP Service Activator VPN User's Guide* for information about Layer 2 Martini VPN devices and data types and for details on the different hardware devices and data encapsulations supported by IP Service Activator for Layer 2 Martini VPNS, and the specific details for VPN types in which there are variations from the typical configuration.

Creating a Sub-interface for a Layer 2 Martini VPN

You create sub-interfaces using configuration policies. See *IP Service Activator VPN User's Guide* for details.

Checking the Configuration Added to the Device

To manually check the configuration added to the device:

1. Access the device through a Telnet session. The sub-interface configuration is added in its own group, called orchestream.
2. Enter the following command:

   ```
   show groups orchestream
   ```

   The sub-interface configuration is displayed.

Hints and Tips

Until you commit your transactions, the created sub-interfaces are not configured on the device. However, you can still use these sub-interfaces in the provisioning of the Layer 2 Martini VPN service. They will be configured, as will the Martini tunnel, when the transactions are committed.

To confirm that the parent interface is capable of supporting the sub-interface you wish to provision:

1. Right-click on the parent interface and select **Properties**
2. On the Interface dialog box, select the Capabilities property page.
3. Under Outbound Properties, expand Martini.
4. Confirm that the type of sub-interface you wish to create to support the Layer 2 Martini VPN is shown.

Devices used in Layer 2 Martini VPNs should be configured to use the **Gateway** role. Interfaces and sub-interfaces used as endpoints should be configured to use the **Access** role.

Deleting Provisioned Sub-interfaces

IP Service Activator does not allow the deletion of a sub-interface if it is part of an existing Layer 2 Martini VPN or is otherwise still in use.

Do not delete sub-interfaces that were created manually (outside of IP Service Activator).

After creating a numbered sub-interface under an interface (in the Orchestream group) on a Juniper router, you can also manually create a same-numbered sub-interface (under the actual interface) on the Juniper router. However, these two sub-interfaces could have different DLCI values.

Use IP Service Activator to create the provisioned sub-interface. This creates a sub-interface number and a DLCI. If you manually create the same-numbered sub-interface on the Juniper router, remember that the DLCI number must be the same as the DLCI for the sub-interface created by IP Service Activator to discover it properly

To delete a provisioned sub-interface:

1. On the Topology tab, open the relevant device, and double click on the parent interface for the sub-interface to be deleted.

   The Details window appears.
2. Select the Provisioned Topology tab.
3. Do one of the following:

- Right click the sub-interface in the Provisioned Topology window, then choose **Delete** from the context menu.
- Choose **Delete** from the Edit menu.
- Click on the Delete button on the toolbar.

Hints and Tips

To remove a sub-interface that was created directly on a device (rather than provisioned using the IP Service Activator client), first log into the device and remove the sub-interface. Then remove the sub-interface from the IP Service Activator client. If you do not first remove the sub-interface from the device, it will re-appear in the IP Service Activator client the next time the device is discovered.

Provisioned sub-interfaces can only be deleted from the Provisioned Topology window as described. The delete button is disabled if trying to delete from another location, for example the hierarchical tree. When you right click a provisioned sub-interface in the hierarchical tree, "Delete" does not appear in the context menu.

Configuration Requirements

This section describes MPLS, OSPF, and LDP features that must be configured in order to support Layer 2 Martini VPNs.

MPLS Requirements

MPLS support must be enabled on all appropriate interfaces.

The command to configure MPLS on all interfaces is made at the [edit protocols] hierarchy level:

```
mpls {
  interface {interface-name | all };
}
```

OSPF Requirements

OSPF (or another IGP) must be configured to support Layer 2 Martini VPNs.

The command to configure OSPF on all interfaces is made at the [edit protocols] hierarchy level:

```
protocols {
  ospf {
    traffic-engineering
    area address {
      interface interface-id
      interface loopback-id
    }
  }
}
```

where *address* is the address of the area, *interface-id* is the interface, and *loopback-id* is the loopback interface.

LDP Requirements

LDP must be configured to support Layer 2 Martini VPNs. On PE devices LSPs must be configured between the loopback addresses of all PE and P devices.

The command to configure LDP is made at the [edit protocols] hierarchy level:

```
protocols {
  ldp {
    interface interface-id
    interface loopback-id
  }
}
```

where *interface-id* is the interface and *loopback-id* is the loopback interface.

Juniper Commands

This section discusses the Juniper commands for Layer 2 Martini VPNs.

Defining Endpoints for Data Encapsulation

The commands to configure endpoints depend on the type of data encapsulation in the Layer 2 Martini VPN.

- Ethernet endpoints: endpoints for Layer 2 Martini VPNs encapsulating Ethernet data are the main interfaces themselves. Therefore, no specific configuration is applied to the device.
- Ethernet VLAN data endpoints: the commands to configure sub-interfaces for Layer 2 Martini VPNs encapsulating Ethernet VLAN data are made at the [edit interfaces] hierarchy level, as follows.

  ```
  interfaces {
    name {
      description "description"
      vlan-tagging;
      encapsulation vlan-ccc;
      unit unit-number {
        encapsulation vlan-ccc;
        vlan-id vlan-id;
      }
    }
  }
  ```

  where *name* is the name of the interface, *description* is a text description of the interface, *unit-number* is the sub-interface number, and *vlan-id* is the VLAN identifier.

- ATM Cell endpoints: the commands to configure endpoints for Layer 2 Martini VPNs encapsulating ATM Cell and ATM AAL5 data are made at the [edit interfaces] and the [edit chassis] hierarchy level.

  At the [edit interfaces] hierarchy level:

  ```
  interfaces {
    name {
      description
      encapsulation atm-ccc-cell-relay;
      atm-options {
        pic-type atm1 | atm2;
        vpi vpi-id;
      }
      unit unit-number {
        encapsulation atm-vc-mux;
        vci vci-id;
  ```

```
    }
  }
}
```

where *name* is the name of the interface, *description* is a text description of the interface, *atm1 | atm2* is the type of PIC card, *vpi-id* is the virtual path identifier, *unit-number* is the sub-interface number, and *vci-id* is the virtual channel identifier.

At the [edit chassis] hierarchy level:

```
chassis {
  fpc fpc-id {
    pic pic-id {
      atm-l2circuit-mode cell | atm-l2circuit-mode aal5;
    }
  }
}
```

where *fpc-id* is the functional processor card, *pic-id* is the processor interface card identifier, and *atm-l2circuit-mode cell | atm-l2circuit-mode aal5* is the type of ATM encapsulation (either ATM cell or rATM AAL5)

- Frame Relay interfaces: the commands to configure endpoints for Layer 2 Martini VPNs encapsulating Frame Relay data follow.

```
interfaces {
  name {
    description
    encapsulation frame-relay-ccc;
    unit unit-number {
      encapsulation frame-relay-ccc;
      dlci dlci-id;
    }
  }
}
```

where *name* is the name of the interface, *description* is a text description of the interface, *unit-number* is the sub-interface number, and *dlci-id* is the data link connection identifier.

Modifying a Sub-interface's DLCI Number

When a sub-interface is part of a Layer 2 Martini connection over frame relay, the DLCI number cannot be modified through the properties dialog box for the sub-interface.

To modify the DLCI number of a sub-interface assigned to a Layer 2 Martini connection:

1. Unlink the sub-interface from the Layer 2 Martini connection by right-clicking on the sub-interface and selecting **Unlink** from the context menu.
2. Click the Provisioned Topology tab in the Details window.
3. Click the sub-interface to be changed.
4. Right-click and select **Properties** from the context menu.
5. Change the DLCI number of the sub-interface.
6. Apply the changes by committing pending transactions.
7. Re-discover the device. The DLCI number is refreshed in the Details window.

8. Drag the sub-interface back to the Layer 2 Martini connection object in order to re-link it to the Martini tunnel.

Working with Pre-existing VLAN, DLCI, and VC Endpoints

In some situations, manually configured VLANs and DLCIs are discovered with the IDs of their parent's logical unit ID rather than with their own ID. This leads to misrepresentation in the client of DLCIs and VLANs IDs when the parent logical unit is provisioned with a different ID than the DLCI or VLAN ID.

For example, a DLCI with ID 567 on logical unit 10 will be discovered as a DLCI with ID 10. When manually provisioning router subinterfaces and virtual circuits, if possible, use matching IDs for logical units and DLCIs or VLANs. For example, for logical unit 567 use a matching DLCI value of 567.

When provisioning Martini circuits (or CCCs) it is possible, that the device driver will generate incorrect router configuration due to issues discovering DLCIs or VLANs.

Create subinterfaces using configuration policies with proper encapsulations and DLCIs or VLANs ID values suitable for planned Martini circuits. Use those subinterfaces as Martini or CCC endpoints rather than manually configured sub-interfaces to avoid any conflicts.

Subinterfaces can be created using configuration policies. Refer to *IP Service Activator VPN User's Guide* for details.

Modifying Endpoint Values

If you change the VLAN/DLCI/VC ID for an endpoint in a Layer 2 Martini VPN, re-discover the device in IP Service Activator.

Defining the Martini Tunnel

The command to define the connection between the endpoints in a Layer 2 Martini VPN is:

```
protocols {
  l2circuit {
    neighbor address {
      description
      interface name {
        virtual-circuit-id virtual-circuit;
      }
    }
  }
}
```

where *address* is the address of the neighbor, *description* is a text description of the Layer 2 Martini VPN, *name* is the name of the interface, and *virtual-circuit* is the identifier for the Layer 2 Martini VPN.

7

Configuring QoS and Access Control Features

This chapter provides detailed information on the Quality of Service (QoS) and access control features configured by Oracle Communications IP Service Activator.

Juniper QoS Features

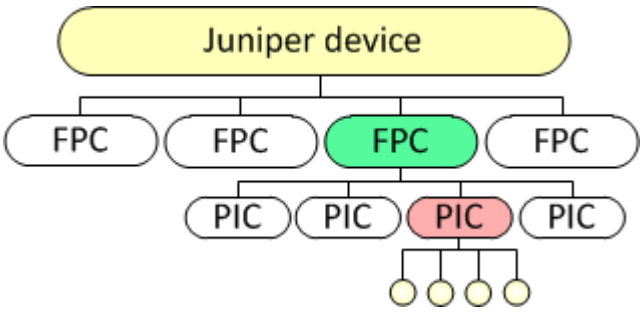
This section describes the key features of the Juniper QoS.

FPC Structure

The Flexible PIC Concentrator (FPC) is a card that slots into the router. FPCs connect the Physical Interface Cards (PICs) to the rest of the router so that incoming packets are forwarded across the midplane to the appropriate destination port. Each FPC holds up to a maximum number of four PICs.

This structure is shown in Figure 7–1.

Figure 7–1 FPC Structure



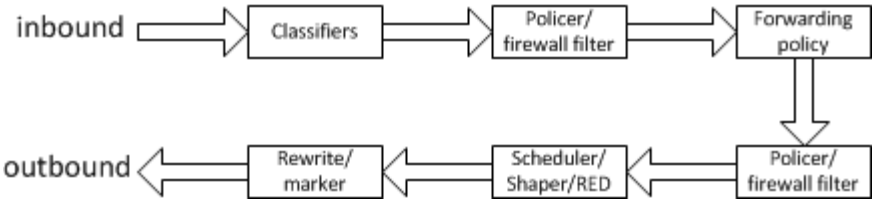
This hardware feature is reflected in the commands that are used to configure the device so that some commands are applied on a per-FPC basis.

For detailed information about the FPC consult the appropriate M-series Hardware guide.

Juniper CoS Overview

Figure 7–2 illustrates the packet flow through the router.

Figure 7–2 CoS Packet Flow



On ingress, the inbound packet is classified and policers and firewall filters may be run to amend the forwarding class and the loss priority. Firewall filters may also direct the forwarding of the packet to an alternative routing instance or interface.

On egress, firewall filters and policers may again be run to modify the forwarding class and the loss priority. The outbound packet is queued according to the forwarding class and scheduling/dropping is applied.

For detailed information on Juniper QoS features, see the JUNOS Software Class of Service Configuration Guide and the JUNOS Software Network Interfaces Configuration Guide, available on the Juniper Networks Technical Documentation Web site:

http://www.juniper.net/techpubs/en_US/release-independent/junos/information-products/pathway-pages/junos/product/

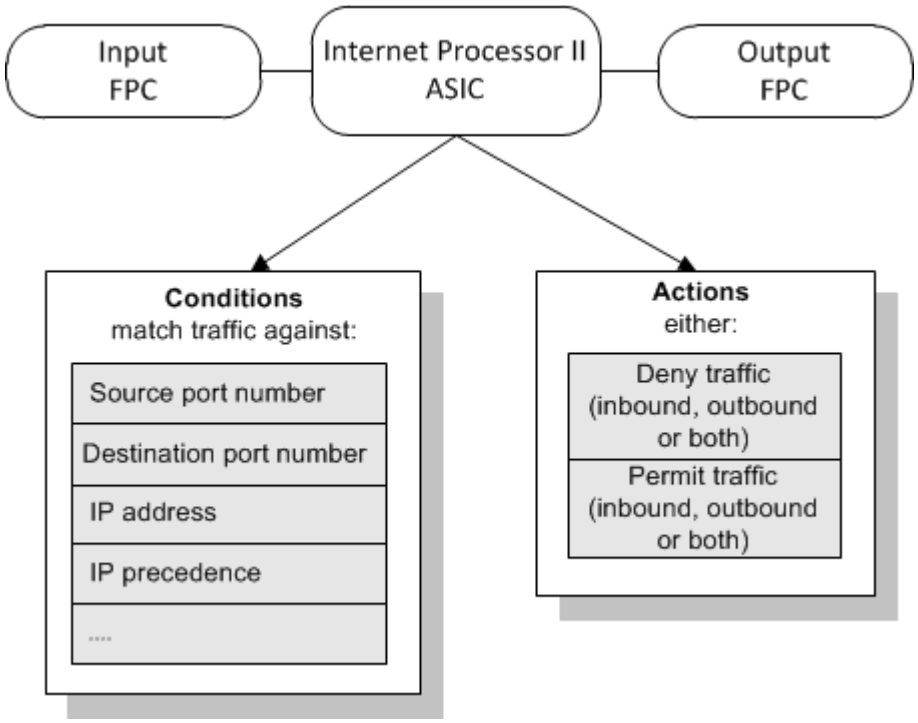
Firewall Filters

A firewall filter classifies packets based on their content and may subsequently perform some action on the classified packets. Each filter comprises one or more filter terms. Each term consists of one or more conditions that define the filtering criteria and the action to take if a match occurs. The ordering of terms within the filter is significant. Packets are tested against each term until a match is found and the associated action is taken. If a packet does not match any of the filter conditions it is silently discarded. It is possible to define a large number of filters without affecting the speed of the interface.

Firewall filters can be applied to inbound or outbound traffic to restrict the packets passing in to or out of the device. Filters can be applied to one, more than one or all interfaces and the same filter can be applied to multiple interfaces.

Filtering is performed within the hardware by the Internet Processor II ASIC. This ASIC is standard on M5, M10, M20, M40 and M160 routers. It can be installed as an upgrade to early versions of M20 and M40 devices which do not have this ASIC.

Figure 7–3 Firewall Filtering



Filter match conditions may be based on:

- Numeric range: such as source or destination port number or IP precedence value
- Addresses: source or destination IP address
- Bit field values: matches can be made on IP options, TCP flags, and IP fragmentation fields

Firewall filters are used by IP Service Activator to implement access rules and rate limiting and to classify traffic for WRR.

The command to configure a firewall filter is made at the [edit firewall family inet] hierarchy level:

```
filter filter-name {
  term term-name {
    from {
      match-conditions;
    }
    then {
      action;
    }
  }
}
```

where *filter-name* is the name of the filter, *term-name* is the name of the filter term, *match-conditions* is the condition that the incoming packets must match for the action to be applied, and *action* is the steps to take for packets that match the filter condition.

Firewall filters are applied to an interface by including the filter statement at the [edit interfaces *interface-name* unit *logical-unit-number* family inet] hierarchy level:

```
interfaces {
  interface-name {
```

```
    unit logical-unit-number {
      family inet {
        filter {
          input filter-name;
          output filter-name;
        }
      }
    }
  }
}
```

where *interface-name* is the name of the interface, *logical-unit-number* is the number of the logical interface, and *filter-name* is the name of the filter to be applied.

For more information see the JUNOS Software Class of Service Configuration Guide and the JUNOS Software Network Interfaces Configuration Guide, available on the Juniper Networks Technical Documentation Web site:

http://www.juniper.net/techpubs/en_US/release-independent/junos/information-products/pathway-pages/junos/product/

Classification of Packets and Queue Selection

On Juniper devices, traffic is classified and enqueued on input to the device. For IP traffic, Juniper devices map incoming traffic to output transmission queues according to the value of the packet header's IP precedence bits or DiffServ codepoint setting. By default, packets are placed in queue 0 with the exception of routing protocol traffic which is placed in queue 3. In IP Service Activator, mapping of an IP precedence bit value to an output queue is indirectly configured when you apply a WRR PHB group.

Access Rules

In IP Service Activator, access rules are used to permit and deny access to identified traffic on a particular interface.

Implementing Access Rules

Access rules are implemented on Juniper devices by firewall filters. Each access rule translates directly to a filter that comprises a single filter term. If an access rule applies to both inbound and outbound traffic, two filters are generated. The generated filters are referenced in the configuration for the interfaces affected by the access rule. Access rules can be applied inbound and outbound.

For firewall filter syntax details, see "Firewall Filters".

The filter name is generated automatically and is in the following format:

`InFilter|OutFilter--`*`interface_name`*

where *interface_name* is the name of the interface.

The filter term name is generated automatically and is in the following format:

`OrchFilterTerm--`*`rule_id`*

where *rule_id* is the object ID of the rule allocated by IP Service Activator.

When access rules are implemented, as well as the specific permit or deny filters, additional filter terms are added to explicitly permit SNMP, Telnet and SSH traffic both

inbound and outbound between the device and the device driver. These terms are always added at the start of the list, and minimize the chance of locking out network control traffic inadvertently.

> **Note:** Starting with version 4.0, IP Service Activator no longer configures an explicit **accept** term to allow any IP traffic not directly matched. The Juniper implicit discard action denies traffic that does not match any of the preceding terms.
>
> Be aware that, depending on your network architecture and use of management interfaces, you may need to appropriately add **accept any** access rules to avoid being locked out of your devices. One approach is to add an accept access rule to devices which is then inherited by the devices' interfaces used for management connectivity.

Features Supported by Access Rules

Access rules may be applied to traffic identified by any of the following methods:

- Source and/or destination IP address
- Source and/or destination port number
- TCP control bits
- IP protocol
- IP precedence value/DiffServ codepoint

> **Note:** Rate limiting (see "Rate Limiting") and WRR classification (see "Weighted Round Robin Queuing Mechanism") are also implemented by firewall filters. If more than one feature is implemented on one interface, terms are automatically ordered so that they have the expected behavior. The order in which they are evaluated is WRR classification terms, followed by rate limiting terms followed by access rule terms.

In addition, header logging is supported in access rules by the **log** command.

Header Logging Configuration Details

Header logging is enabled in the Access Rule property page of the Access Rule dialog box.

The configuration sent to the device if header logging is enabled is:

```
filter name {
  term term-name {
    from {
      other-ACL-options;
    }
    then {
      log;
      {accept|discard};
    }
  }
}
```

About TCP Control Bit Filtering in Access Rules

On Juniper devices, the **tcp-established** keyword is used only when **established** is the only option used.

If other control bits of the TCP header are selected in the access rule, the CLI line is:

```
tcp-flags other-TCP-control-bits & (ack|rst)
```

This is because **tcp-established** is an alias for **tcp-flags ack | rst** and **tcp-flags** commands cannot be repeated in the same filter term.

If TCP options are enabled, the configuration has the following format:

```
filter name {
  term term-name {
    from {
      other-ACL-options;
      [tcp-established;] |
      [tcp-flags boolean-combinations-of-urgent|ack|push|rst|syn|fin;]
    }
    then {
      [log;]
      {accept|discard};
    }
  }
}
```

Example Configuration

This example shows an access rule that has been applied to an access interface with the following configuration:

- Action: Permit
- Direction: In
- Traffic Type: aol-3-c
- Source: Subnet 10.50.0.5
- Destination: 10.50.0.6

The resulting configuration of the access router is as follows:

```
groups {
  orchestream {
    firewall {
      family inet {
        filter InFilter--fe-0-1-3-0 {
          term OrchFilterTerm--3932 {
            from {
              protocol 6;
              source-port 5193;
              source-address 10.50.0.5/32;
              destination-address 10.50.0.6/32;
            }
            then {
              accept;
            }
          }
          }
        }
      }
```

```
    }
    interfaces {
      fe-0/1/3 {
        unit 0 {
          family inet {
            filter {
              input InFilter--fe-0-1-3-0;
            }
          }
        }
      }
    }
  }
}
```

Rate Limiting

Rate limiting enables you to limit the amount of traffic passing in to or out of an interface. It can be used to control access to the core network by constraining specific outbound traffic to a particular bandwidth. Rate limiting is implemented on Juniper routers by a specialized firewall filter.

Rate limiting allows a certain amount of busy traffic before it starts discarding packets. It applies the following rate limits to traffic that enters an interface:

- Bandwidth: the number of permitted bits per second
- Maximum burst size: the maximum burst size, defined in bytes, for traffic that exceeds the maximum bandwidth

Rate limiting is set up within PHB groups.

Rate limiting cannot be applied to MPLS traffic.

Rate limiting can only be applied to routers that have the Internet Processor II ASIC.

Rate limiting is implemented on Juniper devices by a specialized firewall filter. The filter features a policer that filters traffic based on bandwidth and burst size limits and specifies the action to take for traffic exceeding those conditions. Rate limiting is always applied to the outbound interface.

Juniper Commands

The Juniper command to define a policer is:

```
policer policer_name {
  if-exceeding {
    bandwidth-limit rate;
    burst-size-limit bytes;
  }
    then {
    discard;
  }
}
```

where *rate* is the maximum burst rate in bits per second and *bytes* is the maximum number of bytes to be transmitted in bytes per second.

For information on general firewall filter syntax, see "Firewall Filters".

Implementing Rate Limiting

Traffic within a particular class of service is identified by IP precedence bit.

Values input to the client are as follows:

- Average rate: the average transmission rate in Kbits per second for the class of service
- Burst rate: the burst rate in Kbits per second for the class of service
- Burst interval: the interval, in seconds, over which traffic in the selected class of service is allowed to maintain its maximum burst rate

The values required for the rate limiting policer statement are:

- Bandwidth limit: the average number of bits permitted per second
- Burst size limit: the total number of bytes per second in excess of the bandwidth limit allowed in a burst

JUNOS devices calculate burst based on the maximum number of bytes permitted in the burst, rather than the duration of the burst. The Juniper device driver calculates the required values from those input through the client as follows:

- Bandwidth limit (in Kbits per second) = average rate
- Burst size limit (in Kbytes per second) = (burst rate - average rate) * burst interval / 8

Note: Burst rate and average rate must be set to different values.

Example Configuration

This example shows a PHB group that has been configured from the client with the details listed in Table 7–1.

Table 7–1 Example Configuration Details

Class of Service	Average	Burst Rate	Interval
Gold (5)	4000	4500	10
Silver (3)	3000	3500	10
Bronze (0)	2000	2500	10

The resulting configuration is as follows:

```
groups {
  orchestream {
    firewall {
      family inet {
        filter OutFilter--fe-0-1-3-0 {
          policer policer1 {
            if-exceeding {
              bandwidth-limit 2000000;
              burst-size-limit 625;
            }
            then {
              discard;
            }
          }
          policer policer2 {
```

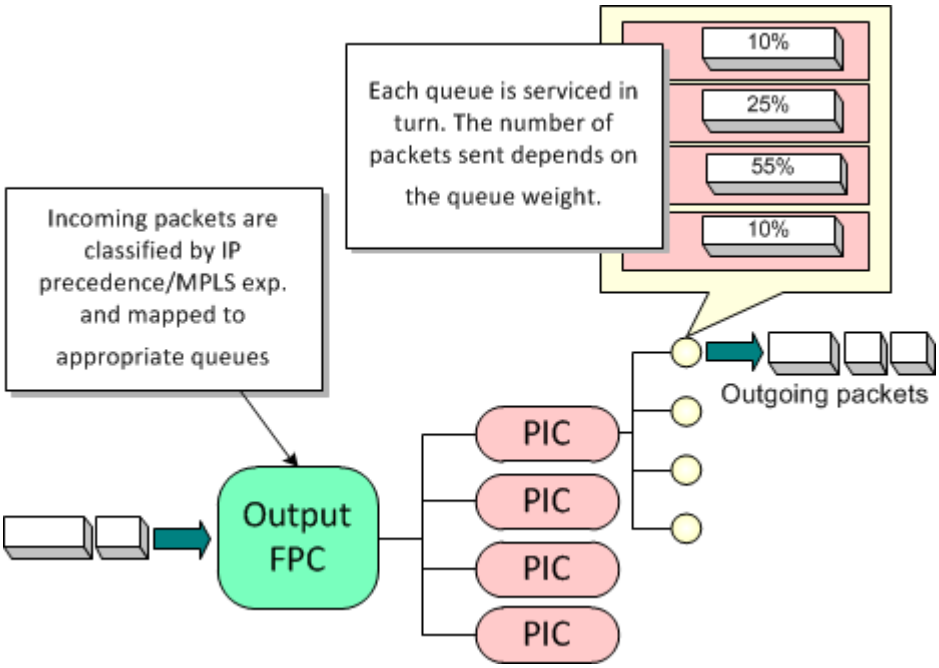
```
            if-exceeding {
              bandwidth-limit 3000000;
              burst-size-limit 625;
            }
            then {
              discard;
            }
          }
          policer policer3 {
            if-exceeding {
              bandwidth-limit 4000000;
              burst-size-limit 625;
            }
            then {
              discard;
            }
          }
          term OrchFilterTerm--policer1 {
            from {
              dscp af33;
            }
            then {
              policer policer1;
              accept;
            }
          }
          term OrchFilterTerm--policer2 {
            from {
              precedence critical-ecp;
            }
            then {
              policer policer2;
                accept;
            }
          }
        term OrchFilterTerm--policer3 {
          from {
            precedence internet-control;
          }
          then {
            policer policer3;
            accept;
          }
        }
      }
    }
    intrfaces {
      fe-0/1/3 {
        unit 0 {
          family inet {
            filter {
              output OutFilter--fe-0-1-3-0;
            }
          }
        }
      }
    }
  }
}
```

Weighted Round Robin Queuing Mechanism

Weighted Round Robin (WRR) allows you to specify the percentage of bandwidth allocated to each queue. A greater number of packets is dequeued from a queue with a higher bandwidth percentage than one with a lower percentage whenever the queue is serviced.

Weights are assigned per queue for each interface within an FPC. Up to four transmission queues can be configured for each output link. The weight is defined as a percentage of the total link transmission bandwidth.

Figure 7–4 Weighted Round Robing Queuing



Only queues containing packets are serviced according to the configured weighting; empty queues are not serviced. This means that a congested queue can borrow buffer space and bandwidth from an under-utilized queue.

It is also possible to configure the memory allocation used by each queue on an interface.

WRR is implemented within IP Service Activator by PHB groups.

For information on traffic classification and allocation of output queues on Juniper devices see "Classification of Packets and Queue Selection".

Juniper Commands

Juniper implements WRR as follows:

- Firewall filters are used to map the traffic to the appropriate forwarding classes
- Scheduling policy maps are used to configure the forwarding classes that represent packet queues and associate them with physical interfaces.

WRR is always applied to outbound traffic.

Classification by Firewall Filters

An outbound firewall filter is used to apply the default mapping from IP precedence to forwarding class.

The format and syntax of the firewall filter command is as described in "Firewall Filters".

Scheduling Policy Maps

A scheduler configuration block is used to specify the bandwidth for a queue. The following Juniper command is configured by IP Service Activator at the [edit class-of-service] hierarchy level:

```
schedulers {
  scheduler-name {
    transmit-rate percent percentage;
  }
}
```

where *scheduler-name* is the name allocated by IP Service Activator in the form Orch-Scheduler-*interface_name* and *percentage* is the calculated percentage of bandwidth to be allocated to the queue.

The following command is used to map a specified forwarding class to a scheduler configuration:

```
scheduler-maps {
  map-name {
    forwarding-class class-name scheduler scheduler-name;
  }
}
```

where *map-name* is the name allocated by IP Service Activator in the form Orch-Scheduler-Map-*interface_name*, *class-name* is the name of the forwarding class (best-effort, expedited-forwarding, assured-forwarding, or network-control), and *scheduler-name* is the name of the associated scheduler.

The scheduler map is then associated with an output interface as follows:

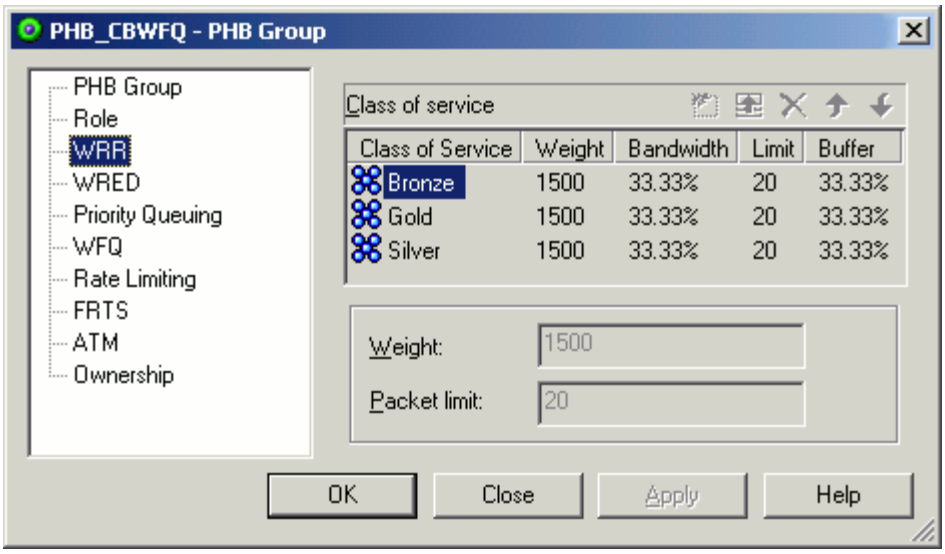
```
interfaces {
  interface-name {
    scheduler-map map-name;
  }
}
```

Implementing Weighted Round Robin Queuing

Traffic within a particular class of service is identified by the value of the IP precedence bits in the packet header or the experimental bits in the MPLS label.

WRR is implemented in IP Service Activator by means of a PHB group, as shown in Figure 7–5.

Figure 7–5 Implementing WRR with a PHB Group



For each class of service, a value can be specified in the weight field, which is converted to a percentage of the available bandwidth. The Packet limit value is not used and any value specified is ignored.

The device driver interprets each value as a ratio from which it calculates the bandwidth percentage. Each percentage value is rounded down to the nearest whole number. For example, if the weights allocated are 3, 6, 10, 1, the driver allocates the following bandwidth percentages: 15%, 30%, 50% and 5%. If the percentage values add up to less than 100% any excess bandwidth is distributed across all queues.

Note: Oracle recommends that routing protocol traffic (IP Precedence 6 and 7 or DiffServ codepoints 48 and 56) is always allocated at least 5% of the available bandwidth.

By default, IP traffic is placed in output transmission queue 0 and network control traffic in queue 3. Queue 0 is assigned 95% of the bandwidth, queue 3 is assigned 5%. If a WRR PHB group is not applied to an interface, this default configuration applies.

Example Juniper MPLS PHB Group-WRR

To support the application of WRR PHBs, IP Service Activator includes a **juniper.policy** file that contains an example Juniper MPLS PHB Group-WRR. This PHB Group lets you define WRR schedulers for four classes of service:

- Juniper-MPLS-best-effort (uses packet marking precedence [0, 1])
- Juniper-MPLS-expedited-forwarding (uses packet marking precedence [2, 3])
- Juniper-MPLS-assured-forwarding (uses packet marking precedence [4, 5])
- Juniper-MPLS-network-control (uses packet marking precedence [6, 7])

The WRR schedulers are applied at the outbound (transmit) direction of the interface and assigned a percentage of interface bandwidth according to configured relative weights. The packet filters of the example Juniper MPLS PHB Group-WRR are implemented as firewall filters and use IP Precedence packet marking to assign packets to one of the 4 forwarding classes:

- precedence [0, 1] (mapped to forwarding-class best-effort)
- precedence [2, 3] (mapped to forwarding-class expedited-forwarding)
- precedence [4, 5] (mapped to forwarding-class assured-forwarding)
- precedence [6, 7] (mapped to forwarding-class network-control)

A limitation of the example Juniper MPLS PHB Group-WRR is that exactly four classes of service as defined in the **juniper.policy** file must be used and these CoSs cannot be modified. Also, the Default CoS visible in the client is ignored by the device driver and can be safely de-selected.

Example Configuration for the Example Juniper MPLS PHB Group-WRR

This example shows the example Juniper MPLS PHB Group-WRR configured from the client with the parameters listed in Table 7–2.

Table 7–2 Example Configuration Parameters

Class of Service	Equivalent Packet Marking	Weight
Juniper-MPLS-best-effort	IP Precedence 0 and 1	25
Juniper-MPLS-expedited-forwarding	IP Precedence 2 and 3	40
Juniper-MPLS-assured-forwarding	IP Precedence 4 and 5	30
Juniper-MPLS-network-control	IP Precedence 6 and 7	5

The resulting router configuration is:

```
orchestream {
  interfaces {
    ge-1/0/0 {
      unit 1 {
        family inet {
          filter {
            output OutFilter-ge-1-0-0-1;
          }
        }
      }
      class-of-service {
        interfaces {
          ge-1/0/0 {
            scheduler-map Orch-Scheduler-Map-ge-1-0-0;
          }
        }
      scheduler-maps {
        Orch-Scheduler-Map-ge-1-0-0 {
          forwarding-class best-effort scheduler Orch-Scheduler-ge-1-0-0-be;
          forwarding-class expedited-forwarding scheduler
Orch-Scheduler-ge-1-0-0-ef;
          forwarding-class assured-forwarding scheduler
Orch-Scheduler-ge-1-0-0-af;
          forwarding-class network-control scheduler Orch-Scheduler-ge-1-0-0-nc;
        }
      }
    schedulers {
      Orch-Scheduler-ge-1-0-0-be {
        transmit-rate percent 25;
      }
      Orch-Scheduler-ge-1-0-0-ef {
```

```
        transmit-rate percent 40;
      }
      Orch-Scheduler-ge-1-0-0-af {
        transmit-rate percent 30;
      }
      Orch-Scheduler-ge-1-0-0-nc {
        transmit-rate percent 5;
      }
    }
  }
firewall {
  family inet {
    filter OutFilter-ge-1-0-0-1 {
      term OrchFilter-term-classifier-0 {
        from {
          precedence [ 0 1 ];
        }
        then {
          forwarding-class best-effort;
          next term;
        }
      }
      term OrchFilter-term-classifier-1 {
        from {
          precedence [ 2 3 ];
        }
        then {
          forwarding-class expedited-forwarding;
          next term;
        }
      }
      term OrchFilter-term-classifier-2 {
        from {
          precedence [ 4 5 ];
        }
        then {
          forwarding-class assured-forwarding;
          next term;
        }
        }
      term OrchFilter-term-classifier-3 {
      from {
        precedence [ 6 7 ];
      }
      then forwarding-class network-control;
    }
  }
```

8

Troubleshooting

This chapter provides troubleshooting guidelines for configuring Juniper M-series devices with Oracle Communications IP Service Activator.

Checking the Juniper M-series Device Logs

The Juniper device driver records all device configuration changes that it makes in an audit log file. You can check these log files to see if configuration is being successfully applied to a device.

A new log file is created following the first transaction after midnight. The same log file is used if a device driver is stopped and started within a 24-hour period.

The log files are created in **/opt/OracleCommunications/ServiceActivator/AuditTrails**.

Each file is named *day***.juniper.audit.log**. For example, the log for a Tuesday would be **Tue.juniper.audit.log**. After a week, the log files are automatically overwritten, so you should archive them within this period if you want to keep them.

Log files are text files, recording the date, time and details of each configuration change made to the devices controlled by the device driver.

Discovering Juniper M-series Devices

It is possible to discover devices in IP Service Activator by specifying an IP address and hop count to discover connected devices within a certain number of hops from the specified device.

The ability to discover devices using this method is dependent on each device's routing information being available in SNMP, the protocol used by IP Service Activator to interrogate routers and network segments. Juniper devices do not provide access to their routing tables through SNMP. Specifying a hop count other than zero therefore does not affect the number of devices discovered.

Note: Oracle recommends that you set the Discovery Method to **Segment scan** in the Discovery dialog box if the devices are known to have large routing tables.

Communication Problems

Ensure that an SNMP Read community is set up on the device. The following command should be included with a suitable community string:

```
[edit snmp]
```

```
community public {
  authorization read-only;
}
```

Commit Errors

Errors or conflicts in the configuration sent to the device are detected by JUNOS when the device driver runs the **commit check** command. The device driver saves the failed configuration to a file named **orchestream_last_failed_config.txt** in the **/var/home/***user_name* directory on the Juniper device.

You can investigate the cause of the failure by entering configuration mode, loading the **orchestream_last_failed_config.txt** file and running the commit check command as follows:

```
hostname> configure
hostname# load override orchestream_last_failed_config.txt
hostname# commit check
```

JUNOS checks the configuration file and displays a message indicating the cause of the error. For example:

```
[edit policy-options community Orch-jm10_2_fe_0_0_1-SOO members origin:0:0]
 invalid autonomous system value at '0' not in range 1 to 65535
error: configuration check-out failed
```

Inheritance Groups

Items in the Base configuration group are inherited from the Orchestream configuration group. However, where there are like-named items found in both groups, the item in the Base configuration takes priority. If configuration you have set up in IP Service Activator is not being activated, you may have a naming collision between the two groups.

Useful JUNOS Commands

There are a number of commands that can be useful in checking device configuration. For full details of the command syntax and an explanation of the reported information, see the Juniper documentation.

Table 8–1 lists some general configuration commands for checking device configuration.

Table 8–1 General Configuration JUNOS Commands

Command	Description
`show configuration`	Displays the configuration that is currently running on the device.
`show chassis hardware`	Indicates whether a device has an Internet Processor or an Internet Processor II ASIC.

Table 8–2 lists monitoring and troubleshooting commands that can be used to display information and statistics about the software and test network connectivity.

Table 8–2 Monitoring and Troubleshooting JUNOS Commands

Command	Description
`clear` *option*	Clears information from various router and protocol databases.
`commit check`	Verifies the syntax of a configuration but does not activate it.
`monitor (start | stop | list)` *interface traffic*	Performs real-time debugging of various software components, including the routing protocols and interfaces.
`ping`	Determines the reachability of a remote network host.
`show` *option*	Displays the current configuration and information about interfaces, routing protocols, routing tables, routing policy filters, and the chassis.
`test` *option*	Tests the configuration and application of policy filters and AS path regular expressions.
`traceroute`	Traces the route to a remote network host.

For detailed information on using these commands see the JUNOS Internet Software Operational Mode Command Reference.

9

Useful References

This chapter provides links to useful pages on the Juniper website and suggestions for further reading.

Juniper Website

Full information about Juniper hardware and software is available from the Juniper website:

http://www.juniper.net/us/en/.

This section highlights some documents that are particularly useful.

Juniper Publications

Index to Juniper publications:

http://www.juniper.net/techpubs/index.html

MPLS Applications PDF (including an overview of CCCs) (PDF, JUNOS 6.0):

http://www.juniper.net/techpubs/software/junos/junos60/swconfig60-mpls-apps/download/swconfig60-mpls-apps.pdf

Overview of QoS features (PDF, JUNOS 6.0)

http://www.juniper.net/techpubs/software/junos/junos60/swconfig60-interfaces/download/swconfig60-interfaces.pdf

Overview of firewall features (PDF, JUNOS 6.0)

http://www.juniper.net/techpubs/software/junos/junos60/swconfig60-policy/download/swconfig60-policy.pdf

Overview of MPLS VPNs (PDF, JUNOS 6.0)

http://www.juniper.net/techpubs/software/junos/junos60/swconfig60-vpns/download/swconfig60-vpns.pdf

Martini Drafts

Current versions of the draft documents describing the Martini protocols implemented in the Juniper M-series Device Driver are available here:

http://www.ietf.org/internet-drafts/?C=S&O=D

The versions specified may not be the latest available at the above link.

Martini, L., et al., “Encapsulation Methods for Transport of Layer 2 Frames Over IP and MPLS Networks”, IETF Internet Draft, draft-martini-l2circuit-encap-mpls-04.txt, work-in-progress.

Martini, L., et al., “Transport of Layer 2 Frames Over MPLS”, IETF Internet Draft, draft-martini-l2circuit-trans-mpls-08.txt, work-in-progress.

Martini, L., et al., “Encapsulation Methods for Transport of Ethernet Frames Over IP and MPLS Networks” IETF Internet Draft, draft-martini-ethernet-encap-mpls-01.txt, work-in-progress.

Martini, L., et al., “Encapsulation Methods for Transport of ATM Cells/Frame Over IP and MPLS Networks”, IETF Internet Draft, draft-martini-atm-encap-mpls-01.txt, work-in-progress.

Juniper M-series Technical Documentation

The following guides are particularly useful:

- JUNOS Software MPLS Applications Configuration Guide

  Provides overview and configuration information for MPLS applications, including CCCs

- JUNOS Software VPNs Configuration Guide

  Provides an overview of Layer 2 and Layer 3 Virtual Private Networks (VPNs), describes how to configure VPNs, and provides configuration examples.

- JUNOS Software Policy Framework Configuration Guide

  Provides an overview of policy concepts and describes how to configure routing policy, firewall filters, and forwarding options.

- JUNOS Software Class of Service Configuration Guide

  Provides an overview of the class of service features and describes how to configure these properties on the routing platform.

- JUNOS Software Network Interfaces Configuration Guide

  Provides an overview of the class of service features and describes how to configure these properties on the routing platform.

A

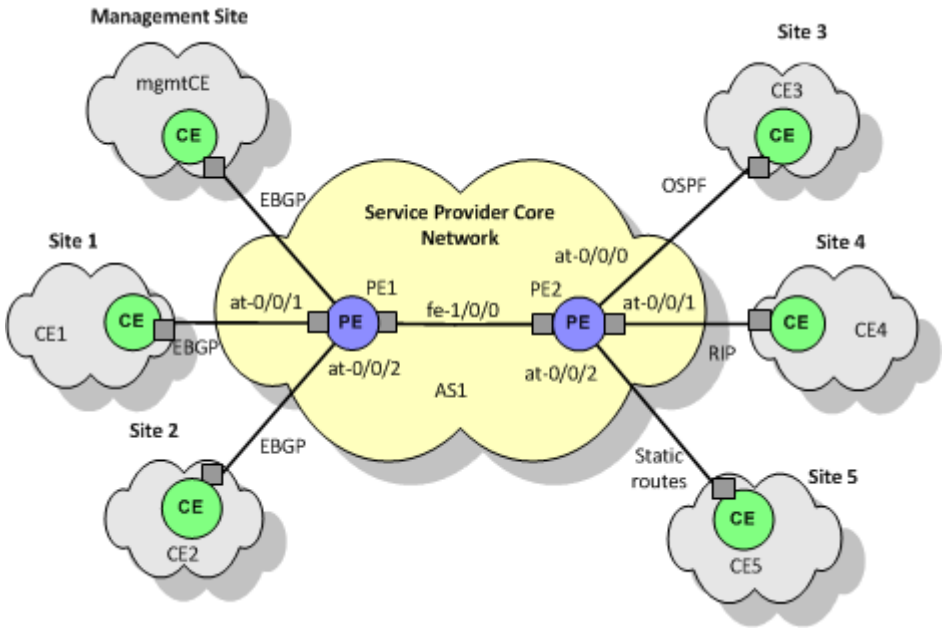
MPLS-VPN Device Configuration

This appendix provides example configuration of the routers involved in an MPLS-VPN.

Sample Network

The network associated with the sample configuration files is shown in Figure A–1.

Figure A–1 Sample Network

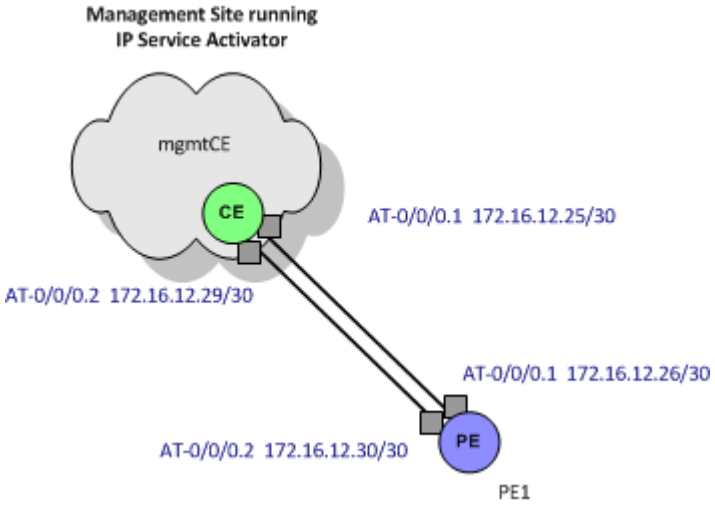


Note the following points:

- A management VPN is set up comprising the Management site and all other sites. With Oracle Communications IP Service Activator running at the management site, all CE routers can be managed.
- Customer VPN 1 comprises sites 1, 2 and 4.
- Customer VPN 2 comprises sites 1, 3 and 5.

In order to configure the management VPN, two links are required between the management site and the core network, one to provide VPN connectivity and one to provide routes to the Service Provider backbone IGP. Figure A–2 illustrates these two links.

Figure A–2 Dual Links Between Management Site and Core Network



The interfaces at both ends of the ATM 0.1 link between PE1 and mgmtCE (the link that is not to be used for the VPN connection) must be assigned a role of Disabled within IP Service Activator to prevent them being configured into a VPN.

Note: For details of the steps required to set up management and customer VPNs, see *IP Service Activator VPN User's Guide*.