Junos® OS

Interchassis Redundancy Using Virtual Chassis Feature Guide for MX Series Routers

Modified: 2017-05-15

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Junos® OS Interchassis Redundancy Using Virtual Chassis Feature Guide for MX Series Routers

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at http://www.juniper.net/support/eula.html. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

About the Documentation xiii
Documentation and Release Notes xiii
Supported Platforms xiii
Using the Examples in This Manual xiii
Merging a Full Example xiv
Merging a Snippet xiv
Documentation Conventions xv
Documentation Feedback xvii
Requesting Technical Support xvii
Self-Help Online Tools and Resources xvii
Opening a Case with JTAC xviii

Chapter 1 **Understanding How Virtual Chassis Provides Interchassis Redundancy 19**

Interchassis Redundancy and Virtual Chassis Overview 19
Interchassis Redundancy Overview 19
Virtual Chassis Overview 20
Benefits of Configuring a Virtual Chassis 20
Supported Routing Platforms for MX Series Virtual Chassis 21
Supported Member Router Combinations 21
Routing Engine Requirements 21

Chapter 2 **Understanding How a Virtual Chassis Works 23**

Virtual Chassis Components Overview 24
Virtual Chassis Master Router 24
Virtual Chassis Backup Router 25
Virtual Chassis Line-Card Router 25
Virtual Chassis Ports 26
Virtual Chassis Port Trunks 27
Slot Numbering in the Virtual Chassis 27
Configuration of Chassis Properties for MPCs in the Virtual Chassis 28
Virtual Chassis Control Protocol 29
Member IDs, Roles, and Serial Numbers 29
Global Roles and Local Roles in a Virtual Chassis 30
Role Name Format 30
Global Role and Local Role Descriptions 31
Mastership Election in a Virtual Chassis 32
Switchover Behavior in an MX Series Virtual Chassis 34
Virtual Chassis Role Transitions During a Global Switchover 34
Virtual Chassis Role Transitions During a Local Switchover 35
Virtual Chassis Role Transitions During Virtual Chassis Formation 36

GRES Readiness in a Virtual Chassis Configuration 37
Split Detection Behavior in a Virtual Chassis 38
How Split Detection Works in a Virtual Chassis 38
Effect of Split Detection on Virtual Chassis Failure Scenarios 38
Virtual Chassis Heartbeat Connection Overview 41
Benefits of Configuring a Virtual Chassis Heartbeat Connection 41
Configuration Requirements for the Heartbeat Connection 42
How the Heartbeat Connection Works 44
Heartbeat Connection and Virtual Chassis Failure Conditions 45
Heartbeat Connection Compared to Split Detection 45
Command Forwarding in a Virtual Chassis 47

Chapter 3 Configuring a Virtual Chassis 57

Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis 58
Preparing for a Virtual Chassis Configuration 60
Installing Junos OS Licenses on Virtual Chassis Member Routers 62
Creating and Applying Configuration Groups for a Virtual Chassis 64
Configuring Preprovisioned Member Information for a Virtual Chassis 66
Configuring Enhanced IP Network Services for a Virtual Chassis 69
Enabling Graceful Routing Engine Switchover and Nonstop Active Routing for a Virtual Chassis 70
Configuring Member IDs for a Virtual Chassis 72
Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis 75
Configuring an MX2020 Member Router in an Existing MX Series Virtual Chassis 92
Switching the Global Master and Backup Roles in a Virtual Chassis Configuration 95
Deleting Member IDs in a Virtual Chassis Configuration 97
Disabling Split Detection in a Virtual Chassis Configuration 98
Example: Replacing a Routing Engine in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers 99
Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers 110
Example: Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers 111
Upgrading an MX Virtual Chassis SCB or SCBE to SCBE2 123
Preparing for the SCBE2 Upgrade 124
Powering Off the MX Series Router 124
Removing an MX Series Routing Engine from an SCB or SCBE 125
Replacing the SCB or SCBE with SCBE2 125
Installing the MX Series Routing Engine into an SCBE2 126
Powering On the MX Series Router 126
Configuring Member IDs for the Virtual Chassis 127
Configuring Virtual Chassis Ports 129
Completing the SCBE2 Upgrade 130

Chapter 4 Configuring Virtual Chassis Ports to Interconnect Member Devices 133
Guidelines for Configuring Virtual Chassis Ports 134
Configuring Virtual Chassis Ports to Interconnect Member Routers or Switches 136
Deleting Virtual Chassis Ports in a Virtual Chassis Configuration 138
Chapter 5 Configuring Locality Bias to Conserve Bandwidth on Virtual Chassis Ports 141
Locality Bias in a Virtual Chassis 141
How Locality Bias Works 141
Locality Bias Percentages 142
Guidelines for Configuring Locality Bias in a Virtual Chassis 143
Configuring Locality Bias for a Virtual Chassis 144
Chapter 6 Configuring Class of Service for Virtual Chassis Ports 145
Class of Service Overview for Virtual Chassis Ports 145
Default CoS Configuration for Virtual Chassis Ports 145
Supported Platforms and Maximums for CoS Configuration of Virtual Chassis Ports 146
Default Classifiers for Virtual Chassis Ports 147
Default Rewrite Rules for Virtual Chassis Ports 147
Default Scheduler Map for Virtual Chassis Ports 148
Customized CoS Configuration for Virtual Chassis Ports 149
Output Traffic-Control Profiles 149
Classifiers and Rewrite Rules 149
Per-Priority Shaping 149
Guidelines for Configuring Class of Service for Virtual Chassis Ports 150
Example: Configuring Class of Service for Virtual Chassis Ports on MX Series 3D Universal Edge Routers 150
Chapter 7 Configuring Redundancy Mechanisms on Aggregated Ethernet Interfaces in a Virtual Chassis 157
Redundancy Mechanisms on Aggregated Ethernet Interfaces in a Virtual Chassis 157
Link Redundancy in a Virtual Chassis 157
Module Redundancy in a Virtual Chassis 158
Chassis Redundancy in a Virtual Chassis 158
Configuring Module Redundancy for a Virtual Chassis 159
Configuring Chassis Redundancy for a Virtual Chassis 161
Multichassis Link Aggregation in a Virtual Chassis 162
Targeted Traffic Distribution on Aggregated Ethernet Interfaces in a Virtual Chassis 163
Targeted Distribution in a Virtual Chassis 163
Benefits of Targeted Distribution 163
Chapter 8 Upgrading Junos OS in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers by Rebooting the Routing Engines 165
Example: Upgrading Junos OS in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers by Rebooting the Routing Engines 165

Chapter 9 Upgrading Junos OS in an MX Series Virtual Chassis by Performing a Unified In-Service Software Upgrade (ISSU) . . . 171

Unified ISSU in a Virtual Chassis . . . 171
Benefits of Performing a Unified ISSU in an Virtual Chassis . . . 171
Prerequisites for Performing a Unified ISSU in a Virtual Chassis . . . 172
How Unified ISSU Works in a Virtual Chassis . . . 172
Virtual Chassis Role Transitions After a Unified ISSU . . . 173
Preparing for a Unified ISSU in an MX Series Virtual Chassis . . . 175
Upgrading Junos OS in an MX Series Virtual Chassis by Performing a Unified ISSU . . . 177

Chapter 10 Monitoring an MX Series Virtual Chassis . . . 181

Accessing the Virtual Chassis Through the Management Interface . . . 182
Verifying the Status of Virtual Chassis Member Routers or Switches . . . 183
Verifying the Operation of Virtual Chassis Ports . . . 183
Verifying Neighbor Reachability for Member Routers or Switches in a Virtual Chassis . . . 184
Verifying Neighbor Reachability for Hardware Devices in a Virtual Chassis . . . 184
Determining GRES Readiness in a Virtual Chassis Configuration . . . 185
Viewing Information in the Virtual Chassis Control Protocol Adjacency Database . . . 186
Viewing Information in the Virtual Chassis Control Protocol Link-State Database . . . 187
Viewing Information About Virtual Chassis Port Interfaces in the Virtual Chassis Control Protocol Database . . . 188
Viewing Virtual Chassis Control Protocol Routing Tables . . . 188
Viewing Virtual Chassis Control Protocol Statistics for Member Devices and Virtual Chassis Ports . . . 189
Verifying and Managing the Virtual Chassis Heartbeat Connection . . . 190
Inline Flow Monitoring for Virtual Chassis Overview . . . 191
Split Detection . . . 191
Syntax of the sampling-instance Statement . . . 192
FPC Slot Numbers for the Virtual Chassis . . . 192
Managing Files on Virtual Chassis Member Routers or Switches . . . 193
Virtual Chassis SNMP Traps . . . 194
Virtual Chassis Slot Number Mapping for Use with SNMP . . . 194
Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet . . . 196
Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets . . . 207

Chapter 11 Tracing Virtual Chassis Operations for Troubleshooting Purposes . . . 223

Tracing Virtual Chassis Operations for MX Series 3D Universal Edge Routers . . . 224
Configuring the Name of the Virtual Chassis Trace Log File . . . 225
Configuring Characteristics of the Virtual Chassis Trace Log File . . . 225
Configuring Access to the Virtual Chassis Trace Log File . . . 226
Using Regular Expressions to Refine the Output of the Virtual Chassis Trace Log File . . . 227
Configuring the Virtual Chassis Operations to Trace . . . 228

Chapter 12 Configuration Statements 231

aggregated-ether-options 232
heartbeat-address (MX Series Virtual Chassis) 234
heartbeat-timeout (MX Series Virtual Chassis) 235
heartbeat-tos (MX Series Virtual Chassis) 236
locality-bias (MX Series Virtual Chassis) 237
logical-interface-chassis-redundancy (MX Series Virtual Chassis) 238
logical-interface-fpc-redundancy (Aggregated Ethernet Subscriber Interfaces) 238
member (MX Series Virtual Chassis) 239
network-services 240
no-split-detection (MX Series Virtual Chassis) 241
preprovisioned (MX Series Virtual Chassis) 242
role (MX Series Virtual Chassis) 243
sampling-instance 244
serial-number (MX Series Virtual Chassis) 245
targeted-distribution (Static Interfaces over Aggregated Ethernet) 246
traceoptions (MX Series Virtual Chassis) 247
virtual-chassis (MX Series Virtual Chassis) 249

Chapter 13 Operational Commands 251

clear virtual-chassis heartbeat (MX Series Virtual Chassis) 252
request system software in-service-upgrade (MX Series 3D Universal Edge Routers and EX9200 Switches) 254
request virtual-chassis member-id delete (MX Series Virtual Chassis) 271
request virtual-chassis member-id set 272
request virtual-chassis routing-engine master switch 274
request virtual-chassis vc-port delete (MX Series Virtual Chassis) 276
request virtual-chassis vc-port set (MX Series Virtual Chassis) 278
show chassis network-services 280
show interfaces vcp 282
show services accounting status 300
show virtual-chassis active-topology (MX Series Virtual Chassis) 303
show virtual-chassis device-topology (MX Series Virtual Chassis) 305
show virtual-chassis heartbeat (MX Series Virtual Chassis) 308
show virtual-chassis protocol adjacency (MX Series Virtual Chassis) 311
show virtual-chassis protocol database (MX Series Virtual Chassis) 315
show virtual-chassis protocol interface (MX Series Virtual Chassis) 321
show virtual-chassis protocol route (MX Series Virtual Chassis) 324
show virtual-chassis protocol statistics (MX Series Virtual Chassis) 327
show virtual-chassis status (MX Series Virtual Chassis) 330
show virtual-chassis vc-port (MX Series Virtual Chassis) 332

List of Figures

Chapter 2 **Understanding How a Virtual Chassis Works 23**

Figure 1: Sample Topology for MX Series Virtual Chassis 24

Chapter 3 **Configuring a Virtual Chassis 57**

Figure 2: Sample Topology for a Virtual Chassis with Two MX Series Routers . . . 76
Figure 3: Sample Topology for a Virtual Chassis with Two MX Series Routers . . . 101
Figure 4: Sample Topology for a Virtual Chassis with Two MX Series Routers . . . 113

Chapter 8 **Upgrading Junos OS in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers by Rebooting the Routing Engines 165**

Figure 5: Sample Topology for a Virtual Chassis with Two MX Series Routers . . 166

List of Tables

About the Documentation xiii

Table 1: Notice Icons xv
Table 2: Text and Syntax Conventions xvi

Chapter 1 Understanding How Virtual Chassis Provides Interchassis Redundancy 19

Table 3: MX Series Virtual Chassis Supported Member Router Combinations 21

Chapter 2 Understanding How a Virtual Chassis Works 23

Table 4: Slot Count and Slot Numbering for MX Series Virtual Chassis Supported Member Routers 28
Table 5: Global Roles and Local Roles 31
Table 6: Virtual Chassis Role Transitions During Global Switchover 35
Table 7: Virtual Chassis Role Transitions During Local Switchover Performed from VC-Mm 35
Table 8: Virtual Chassis Role Transitions During Local Switchover Performed from VC-Bm 36
Table 9: Effect of Split Detection on Common Virtual Chassis Failure Scenarios 39
Table 10: Comparison of Heartbeat Connection Configuration Tasks for Member Routers in Same Subnet and Member Routers in Different Subnets 44
Table 11: Effect of Heartbeat Connection on Common Virtual Chassis Failure Conditions 45
Table 12: Comparison of Heartbeat Connection and Split Detection for Virtual Chassis Failure Conditions 46
Table 13: Commands Available for Command Forwarding in an MX Series Virtual Chassis 47

Chapter 3 Configuring a Virtual Chassis 57

Table 14: Components of the Sample MX Series Virtual Chassis 77
Table 15: Virtual Chassis Global Role Transitions Before and After Mastership Switchover 96
Table 16: Components of the Sample MX Series Virtual Chassis 101
Table 17: Virtual Chassis Role Transitions Before and After Local Routing Engine Switchover 109
Table 18: Components of the Sample MX Series Virtual Chassis 113

Chapter 6 Configuring Class of Service for Virtual Chassis Ports 145

Table 19: Sample CoS Scheduler Hierarchy for Virtual Chassis Ports 151

Chapter 8 Upgrading Junos OS in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers by Rebooting the Routing Engines 165

Table 20: Components of the Sample MX Series Virtual Chassis 167

Chapter 9 **Upgrading Junos OS in an MX Series Virtual Chassis by Performing a Unified In-Service Software Upgrade (ISSU) 171**

Table 21: Virtual Chassis Role Transitions After Unified ISSU 173

Chapter 10 **Monitoring an MX Series Virtual Chassis 181**

Table 22: jnxFruSlot Numbers and Corresponding Slot Numbers in an MX Series or EX9200 Virtual Chassis 195
Table 23: Components of the Sample MX Series Virtual Chassis with Member Routers in Same Subnet 198
Table 24: Heartbeat Cross-Connections for Member Routers in Different Subnets 209
Table 25: Components of the Sample MX Series Virtual Chassis with Member Routers in Different Subnets 210

Chapter 11 **Tracing Virtual Chassis Operations for Troubleshooting Purposes 223**

Table 26: Tracing Flags for Virtual Chassis 228

Chapter 13 **Operational Commands 251**

Table 27: clear virtual-chassis heartbeat Output Fields 253
Table 28: show chassis network services Output Fields 280
Table 29: show interfaces vcp Output Fields 283
Table 30: show services accounting status Output Fields 300
Table 31: show virtual-chassis active-topology Output Fields 303
Table 32: show virtual-chassis device-topology Output Fields 305
Table 33: show virtual-chassis heartbeat Output Fields 309
Table 34: show virtual-chassis protocol adjacency Output Fields 312
Table 35: show virtual-chassis protocol database Output Fields 316
Table 36: show virtual-chassis protocol interface Output Fields 322
Table 37: show virtual-chassis protocol route Output Fields 324
Table 38: show virtual-chassis protocol statistics Output Fields 327
Table 39: show virtual-chassis status Output Fields 330
Table 40: show virtual-chassis vc-port Output Fields 332

About the Documentation

- Documentation and Release Notes on page xiii
- Supported Platforms on page xiii
- Using the Examples in This Manual on page xiii
- Documentation Conventions on page xv
- Documentation Feedback on page xvii
- Requesting Technical Support on page xvii

Documentation and Release Notes

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at http://www.juniper.net/techpubs/.

If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at http://www.juniper.net/books.

Supported Platforms

For the features described in this document, the following platforms are supported:

- MX Series

Using the Examples in This Manual

If you want to use the examples in this manual, you can use the **load merge** or the **load merge relative** command. These commands cause the software to merge the incoming configuration into the current candidate configuration. The example does not become active until you commit the candidate configuration.

If the example configuration contains the top level of the hierarchy (or multiple hierarchies), the example is a *full example*. In this case, use the **load merge** command.

If the example configuration does not start at the top level of the hierarchy, the example is a *snippet*. In this case, use the **load merge relative** command. These procedures are described in the following sections.

Merging a Full Example

To merge a full example, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration example into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following configuration to a file and name the file **ex-script.conf**. Copy the **ex-script.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   system {
     scripts {
       commit {
         file ex-script.xsl;
       }
     }
   }
   interfaces {
     fxp0 {
       disable;
       unit 0 {
         family inet {
           address 10.0.0.1/24;
         }
       }
     }
   }
   ```

2. Merge the contents of the file into your routing platform configuration by issuing the **load merge** configuration mode command:

   ```
   [edit]
   user@host# load merge /var/tmp/ex-script.conf
   load complete
   ```

Merging a Snippet

To merge a snippet, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration snippet into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following snippet to a file and name the file **ex-script-snippet.conf**. Copy the **ex-script-snippet.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   commit {
     file ex-script-snippet.xsl; }
   ```

2. Move to the hierarchy level that is relevant for this snippet by issuing the following configuration mode command:

```
[edit]
user@host# edit system scripts
[edit system scripts]
```

3. Merge the contents of the file into your routing platform configuration by issuing the **load merge relative** configuration mode command:

```
[edit system scripts]
user@host# load merge relative /var/tmp/ex-script-snippet.conf
load complete
```

For more information about the **load** command, see CLI Explorer.

Documentation Conventions

Table 1 on page xv defines notice icons used in this guide.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.
	Tip	Indicates helpful information.
	Best practice	Alerts you to a recommended use or implementation.

Table 2 on page xvi defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents text that you type.	To enter configuration mode, type the **configure** command: user@host> **configure**
`Fixed-width text like this`	Represents output that appears on the terminal screen.	`user@host>` **show chassis alarms** `No alarms currently active`
Italic text like this	• Introduces or emphasizes important new terms. • Identifies guide names. • Identifies RFC and Internet draft titles.	• A policy *term* is a named structure that defines match conditions and actions. • *Junos OS CLI User Guide* • RFC 1997, *BGP Communities Attribute*
Italic text like this	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: [edit] root@# **set system domain-name** *domain-name*
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	• To configure a stub area, include the **stub** statement at the **[edit protocols ospf area area-id]** hierarchy level. • The console port is labeled **CONSOLE**.
< > (angle brackets)	Encloses optional keywords or variables.	**stub <default-metric** *metric***>;**
| (pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	**broadcast | multicast** **(***string1* **|** *string2* **|** *string3***)**
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	**rsvp { # Required for dynamic MPLS only**
[ ] (square brackets)	Encloses a variable for which you can substitute one or more values.	**community name members [** *community-ids* **]**

Indention and braces ({ })	Identifies a level in the configuration hierarchy.	[edit]
		routing-options {
		  static {
		  route default {
		  nexthop *address*;
		  retain;
		  }
		  }
		}

; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	

GUI Conventions

Table 2: Text and Syntax Conventions *(continued)*

Convention	Description	Examples
Bold text like this	Represents graphical user interface (GUI) items you click or select.	• In the Logical Interfaces box, select **All Interfaces**. • To cancel the configuration, click **Cancel**.
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select **Protocols>Ospf**.

Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can provide feedback by using either of the following methods:

- Online feedback rating system—On any page of the Juniper Networks TechLibrary site at http://www.juniper.net/techpubs/index.html, simply click the stars to rate the content, and use the pop-up form to provide us with information about your experience. Alternately, you can use the online feedback form at http://www.juniper.net/techpubs/feedback/.
- E-mail—Send your comments to techpubs-comments@juniper.net. Include the document or topic name, URL or page number, and software version (if applicable).

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or Partner Support Service support contract, or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf.
- Product warranties—For product warranty information, visit http://www.juniper.net/support/warranty/.
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: http://www.juniper.net/customers/support/
- Search for known bugs: http://www2.juniper.net/kb/
- Find product documentation: http://www.juniper.net/techpubs/
- Find solutions and answer questions using our Knowledge Base: http://kb.juniper.net/
- Download the latest versions of software and review release notes: http://www.juniper.net/customers/csc/software/
- Search technical bulletins for relevant hardware and software notifications: http://kb.juniper.net/InfoCenter/
- Join and participate in the Juniper Networks Community Forum: http://www.juniper.net/company/communities/
- Open a case online in the CSC Case Management tool: http://www.juniper.net/cm/

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: https://tools.juniper.net/SerialNumberEntitlementSearch/

Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at http://www.juniper.net/cm/.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see http://www.juniper.net/support/requesting-support.html.

CHAPTER 1

Understanding How Virtual Chassis Provides Interchassis Redundancy

- Interchassis Redundancy and Virtual Chassis Overview on page 19

Interchassis Redundancy and Virtual Chassis Overview

As more high-priority voice and video traffic is carried on the network, interchassis redundancy has become a baseline requirement for providing stateful redundancy on broadband subscriber management equipment such as broadband services routers, broadband network gateways, and broadband remote access servers. To provide a stateful interchassis redundancy solution for MX Series 3D Universal Edge Routers, you can configure a Virtual Chassis.

This topic provides an overview of interchassis redundancy and the Virtual Chassis, and explains the benefits of configuring a Virtual Chassis on supported MX Series routers.

- Interchassis Redundancy Overview on page 19
- Virtual Chassis Overview on page 20
- Benefits of Configuring a Virtual Chassis on page 20
- Supported Routing Platforms for MX Series Virtual Chassis on page 21

Interchassis Redundancy Overview

Traditionally, redundancy in broadband edge equipment has used an intrachassis approach, which focuses on providing redundancy within a single system. However, a single-system redundancy mechanism no longer provides the degree of high availability required by service providers who must carry mission-critical voice and video traffic on their network. Consequently, service providers are requiring interchassis redundancy solutions that can span multiple systems that are colocated or geographically dispersed.

Interchassis redundancy is a high availability feature that prevents network outages and protects routers against access link failures, uplink failures, and wholesale chassis failures without visibly disrupting the attached subscribers or increasing the network management burden for service providers. Network outages can cause service providers to lose revenues and require them to register formal reports with government agencies. A robust interchassis redundancy implementation enables service providers to fulfill strict

service-level agreements (SLAs) and avoid unplanned network outages to better meet the needs of their customers.

Virtual Chassis Overview

One approach to providing interchassis redundancy is the Virtual Chassis model. In general terms, a *Virtual Chassis* configuration enables a collection of member routers to function as a single virtual router, and extends the features available on a single router to the member routers in the Virtual Chassis. The interconnected member routers in a Virtual Chassis are managed as a single network element that appears to the network administrator as a single chassis with additional line card slots, and to the access network as a single system.

To provide a stateful interchassis redundancy solution for MX Series 3D Universal Edge Routers, you can configure a Virtual Chassis. An MX Series Virtual Chassis interconnects two MX Series routers into a logical system that you can manage as a single network element. The member routers in a Virtual Chassis are designated as the *Virtual Chassis master router* (also known as the *protocol master*) and the *Virtual Chassis backup router* (also known as the *protocol backup*). The member routers are interconnected by means of dedicated *Virtual Chassis ports* that you configure on Modular Port Concentrator/Modular Interface Card (MPC/MIC) interfaces.

An MX Series Virtual Chassis is managed by the *Virtual Chassis Control Protocol* (*VCCP*), which is a dedicated control protocol based on IS-IS. VCCP runs on the Virtual Chassis port interfaces and is responsible for building the Virtual Chassis topology, electing the Virtual Chassis master router, and establishing the interchassis routing table to route traffic within the Virtual Chassis.

NOTE: MX Series Virtual Chassis does not support Ethernet OAM, distributed inline connectivity fault management, Ethernet frame delay measurement, loss measurement, synthetic loss measurement, and Ethernet alarm indication signal (ETH-AIS).

Benefits of Configuring a Virtual Chassis

Configuring a Virtual Chassis for MX Series routers provides the following benefits:

- Simplifies network management of two routers that are either colocated or geographically dispersed across a Layer 2 point-to-point network.
- Provides resiliency against network outages and protects member routers against access link failures, uplink failures, and chassis failures without visibly disrupting attached subscribers or increasing the network management burden for service providers.
- Extends the high availability capabilities of applications such as graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) beyond a single MX Series router to both member routers in the Virtual Chassis.

- Enables service providers to fulfill strict service level agreements (SLAs) and avoid unplanned network outages to better meet their customers' needs.
- Provides the ability to scale bandwidth and service capacity as more high-priority voice and video traffic is carried on the network.

Supported Routing Platforms for MX Series Virtual Chassis

You can configure a Virtual Chassis on the following MX Series 3D Universal Edge Routers with MPC/MIC interfaces:

- MX240 3D Universal Edge Router
- MX480 3D Universal Edge Router
- MX960 3D Universal Edge Router
- MX2010 3D Universal Edge Router
- MX2020 3D Universal Edge Router

NOTE: **Platform support depends on the Junos OS release in your installation.**

Graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) must be enabled on both member routers in the Virtual Chassis.

Supported Member Router Combinations

A two-member MX Series Virtual Chassis supports the member router combinations marked as Yes in Table 3 on page 21.

Table 3: MX Series Virtual Chassis Supported Member Router Combinations

Member Router Type	MX240	MX480	MX960	MX2010	MX2020
MX240	Yes	Yes	Yes	No	No
MX480	Yes	Yes	Yes	No	No
MX960	Yes	Yes	Yes	Yes	Yes
MX2010	No	No	Yes	Yes	Yes
MX2020	No	No	Yes	Yes	Yes

Routing Engine Requirements

Each member router in the Virtual Chassis must have dual Routing Engines installed, and all four Routing Engines in the Virtual Chassis must be the same model. For example, you cannot configure a Virtual Chassis if one member router has two RE-S-2000 Routing

Engines installed and the other member router has two RE-S-1800 Routing Engines installed.

NOTE: For an MX Series Virtual Chassis configuration that includes an MX2020 router, all four Routing Engines in the Virtual Chassis must have at least 16 gigabytes of memory.

Related Documentation

- Virtual Chassis Components Overview on page 24
- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

CHAPTER 2

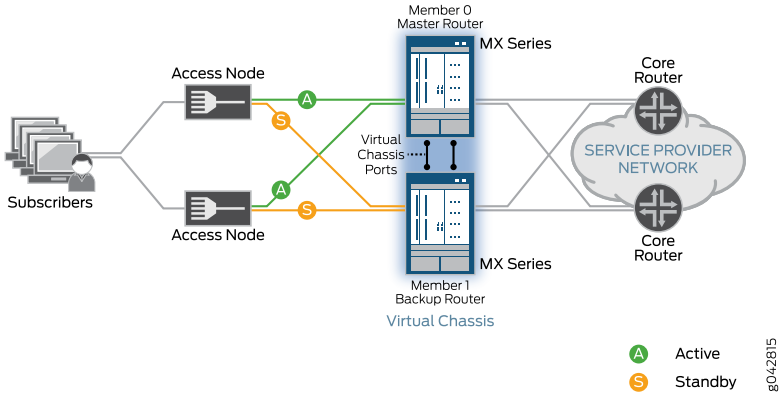
Understanding How a Virtual Chassis Works

- Virtual Chassis Components Overview on page 24
- Global Roles and Local Roles in a Virtual Chassis on page 30
- Mastership Election in a Virtual Chassis on page 32
- Switchover Behavior in an MX Series Virtual Chassis on page 34
- Split Detection Behavior in a Virtual Chassis on page 38
- Virtual Chassis Heartbeat Connection Overview on page 41
- Command Forwarding in a Virtual Chassis on page 47

Virtual Chassis Components Overview

A Virtual Chassis configuration for MX Series 3D Universal Edge Routers interconnects two MX Series routers into a logical system that you can manage as a single network element.Figure 1 on page 24 illustrates a typical topology for a two-member MX Series Virtual Chassis.

Figure 1: Sample Topology for MX Series Virtual Chassis



This overview describes the basic hardware and software components of the Virtual Chassis configuration illustrated in Figure 1 on page 24, and covers the following topics:

- Virtual Chassis Master Router on page 24
- Virtual Chassis Backup Router on page 25
- Virtual Chassis Line-Card Router on page 25
- Virtual Chassis Ports on page 26
- Virtual Chassis Port Trunks on page 27
- Slot Numbering in the Virtual Chassis on page 27
- Configuration of Chassis Properties for MPCs in the Virtual Chassis on page 28
- Virtual Chassis Control Protocol on page 29
- Member IDs, Roles, and Serial Numbers on page 29

Virtual Chassis Master Router

One of the two member routers in the Virtual Chassis becomes the *master router*, also known as the *protocol master*. The Virtual Chassis master router maintains the global configuration and state information for both member routers, and runs the chassis management processes. The master Routing Engine that resides in the Virtual Chassis master router becomes the global master for the Virtual Chassis.

Specifically, the master Routing Engine that resides in the Virtual Chassis master router performs the following functions in a Virtual Chassis:

- Manages both the master and backup member routers
- Runs the chassis management processes and control protocols
- Receives and processes all incoming and exception path traffic destined for the Virtual Chassis
- Propagates the Virtual Chassis configuration (including member IDs, roles, and configuration group definitions and applications) to the members of the Virtual Chassis

The first member of the Virtual Chassis becomes the initial master router by default. After the Virtual Chassis is formed with both member routers, the Virtual Chassis Control Protocol (VCCP) software runs a mastership election algorithm to elect the master router for the Virtual Chassis configuration.

NOTE: **You cannot configure mastership election for an MX Series Virtual Chassis in the current release.**

Virtual Chassis Backup Router

The member router in the Virtual Chassis that is not designated as the master router becomes the *backup router*, also known as the *protocol backup*. The Virtual Chassis backup router takes over mastership of the Virtual Chassis if the master router is unavailable, and synchronizes routing and state information with the master router. The master Routing Engine that resides in the Virtual Chassis backup router becomes the global backup for the Virtual Chassis.

Specifically, the master Routing Engine that resides in the Virtual Chassis backup router performs the following functions in a Virtual Chassis:

- If the master router fails or is unavailable, takes over mastership of the Virtual Chassis in order to preserve routing information and maintain network connectivity without disruption
- Synchronizes routing and application state, including routing tables and subscriber state information, with the master Routing Engine that resides in the Virtual Chassis master router
- Relays chassis control information, such as line card presence and alarms, to the master router

Virtual Chassis Line-Card Router

NOTE: **The line-card role is not supported in the preprovisioned configuration for a two-member MX Series Virtual Chassis. In this release, the line-card role applies only in the context of split detection behavior.**

A member router functioning in the **line-card** role runs only a minimal set of chassis management processes required to relay chassis control information, such as line card presence and alarms, to the Virtual Chassis master router.

You cannot explicitly configure a member router with the **line-card** role in the current release. However, if the backup router fails in a two-member Virtual Chassis configuration and split detection is enabled (the default behavior), the master router takes a **line-card** role, and line cards (FPCs) that do not host Virtual Chassis ports go offline. This state effectively isolates the master router and removes it from the Virtual Chassis until connectivity is restored. As a result, routing is halted and the Virtual Chassis configuration is disabled.

Virtual Chassis Ports

Virtual Chassis ports are special Ethernet interfaces that form a point-to-point connection between the member routers in a Virtual Chassis. When you create a Virtual Chassis, you must configure the Virtual Chassis ports on Modular Port Concentrator/Modular Interface Card (MPC/MIC) interfaces. After you configure a Virtual Chassis port, it is renamed **vcp-*slot/pic/port*** (for example, **vcp-2/2/0**), and the line card associated with that port comes online. For example, the sample Virtual Chassis topology shown in Figure 1 on page 24 has a total of four Virtual Chassis ports (represented by the blue dots), two on each of the two member routers.

After a Virtual Chassis port is configured, it is dedicated to the task of interconnecting member routers, and is no longer available for configuration as a standard network port. To restore this port to the global configuration and make it available to function as a standard network port, you must delete the Virtual Chassis port from the Virtual Chassis configuration.

NOTE: **The Junos OS software enables you to preconfigure ports that are currently unavailable for use. Although a Virtual Chassis port is unavailable for use as a standard network port, you can configure this port as a standard network port even after you configure it as a Virtual Chassis port. However, the router does not apply the configuration until you delete the Virtual Chassis port from the Virtual Chassis configuration.**

You can configure a Virtual Chassis port on either a 1-Gigabit Ethernet (ge) interface, a 10-Gigabit Ethernet (xe) interface, a 40-Gigabit Ethernet (et) interface, or a 100-Gigabit Ethernet (et) interface. 40-Gigabit and 100-Gigabit Virtual Chassis ports can only be configured on MPC3, MPC4, or later line cards. (Interface support depends on the Junos OS release in your installation.) You cannot configure a combination of 1-Gigabit Ethernet Virtual Chassis ports and 10-Gigabit Ethernet Virtual Chassis ports in the same Virtual Chassis. We recommend that you configure Virtual Chassis ports only on 10-Gigabit Ethernet interfaces. In addition, to minimize network disruption in the event of a router or link failure, configure redundant Virtual Chassis ports that reside on different line cards in each member router.

Virtual Chassis port interfaces carry both VCCP packets and internal control and data traffic. Because the internal control traffic is neither encrypted nor authenticated, make

sure the Virtual Chassis port interfaces are properly secured to prevent malicious third-party attacks on the data.

Virtual Chassis ports use a default class of service (CoS) configuration that applies equally to all Virtual Chassis port interfaces configured in a Virtual Chassis. Optionally, you can create a customized CoS traffic-control profile and apply it to all Virtual Chassis port interfaces. For example, you might want to create a nondefault traffic-control profile that allocates more than the default 5 percent of the Virtual Chassis port bandwidth to control traffic, or that assigns different priorities and excess rates to different forwarding classes.

Virtual Chassis Port Trunks

If two or more Virtual Chassis ports of the same type and speed are configured between the same two member routers in an MX Series Virtual Chassis, the Virtual Chassis Control Protocol (VCCP) bundles these Virtual Chassis port interfaces into a trunk, reduces the routing cost accordingly, and performs traffic load balancing across all of the Virtual Chassis port interfaces (also referred to as Virtual Chassis port links) in the trunk.

A Virtual Chassis port trunk must include only Virtual Chassis ports of the same type and speed. For example, a Virtual Chassis port trunk can include either all 10-Gigabit Ethernet (xe media type) Virtual Chassis ports or all 1-Gigabit Ethernet (ge media type) Virtual Chassis ports. An MX Series Virtual Chassis does *not* support a combination of 1-Gigabit Ethernet Virtual Chassis ports and 10-Gigabit Ethernet Virtual Chassis ports in the same Virtual Chassis port trunk.

The router uses the following formula to determine the cost metric of a Virtual Chassis port link in a Virtual Chassis port trunk:

Cost = (300 * 1,000,000,000) / *port-speed*

where *port-speed* is the aggegate speed, in bits per second, of the Virtual Chassis port.

For example, a 10-Gigabit Ethernet Virtual Chassis port link has a cost metric of 30 (300 * 1,000,000,000 / 10,000,000,000). A 1-Gigabit Ethernet Virtual Chassis port link has a cost metric of 300 (300 * 1,000,000,000 / 1,000,000,000). Virtual Chassis port links with a lower cost metric are preferred over those with a higher cost metric.

An MX Series Virtual Chassis supports up to 16 Virtual Chassis ports per trunk.

Slot Numbering in the Virtual Chassis

After you configure the member ID and, optionally, slot count for each router that you want to add to an MX Series Virtual Chassis, the Routing Engines in that chassis reboot and the slots for line cards (FPCs) are renumbered. The FPC slot numbering used for each member router is based on the slot count and offsets used in the Virtual Chassis instead of the physical slot numbers where the line card is actually installed.

Table 4 on page 28 shows the valid slot count values for each supported member router type, and the slot numbering used for member 0 and member 1 when the specified slot count value is configured, either explicitly or by default.

Table 4: Slot Count and Slot Numbering for MX Series Virtual Chassis Supported Member Routers

Member Router Type	Slot Count	FPC Slot Numbers on member 0	FPC Slot Numbers on member 1
MX240	N/A	0 through 11 (no offset)	12 through 23 (offset=12)
MX480	N/A	0 through 11 (no offset)	12 through 23 (offset=12)
MX960	12 (default)	0 through 11 (no offset)	12 through 23 (offset=12)
MX960	20	0 through 19 (no offset)	20 through 39 (offset=20)
MX2010	12 (default)	0 through 11 (no offset)	12 through 23 (offset=12)
MX2010	20	0 through 19 (no offset)	20 through 39 (offset=20)
MX2020	20 (default)	0 through 19 (no offset)	20 through 39 (offset=20)

For example, assume that in your Virtual Chassis configuration, member 0 is an MX960 router and member 1 is an MX2010 router, with the default slot count (12) in effect on both routers. In this topology, a 10-Gigabit Ethernet interface that appears as xe-14/2/2 (FPC slot 14, PIC slot 2, port 2) in the **show interfaces** command output is actually physical interface xe-2/2/2 (FPC slot 2, PIC slot 2, port 2) on member 1 after deducting the offset of 12 for member 1.

Building on this example, assume that you replace member 1 with an MX2020 member router, resulting in a Virtual Chassis with an MX960 router configured as member 0 and an MX2020 router configured as member 1. To ensure that a Virtual Chassis consisting of an MX2020 router and either an MX960 router or MX2010 router forms properly, you must explicitly set the slot count for the MX960 router or MX2010 router to 20 to match the slot count of the MX2020 router. When the FPC slots are renumbered in this topology, physical interface xe-2/2/2 on member 1 becomes xe-22/2/2 on member 1 after adding the offset of 20 for member 1. Similarly, the **show interfaces** command displays xe-22/2/2 as the interface name.

NOTE: Slot renumbering does not affect the names of Virtual Chassis ports. The Virtual Chassis port name, in the format vcp-*slot/pic/port*, is derived from the physical slot number where the port is configured. For example, vcp-3/2/0 is configured on FPC physical slot 3, PIC slot 2, port 0.

Configuration of Chassis Properties for MPCs in the Virtual Chassis

When you configure chassis properties for MPCs installed in a member router in an MX Series Virtual Chassis, keep the following points in mind:

- Statements included at the **[edit chassis member** *member-id* **fpc slot** *slot-number***]** hierarchy level apply to the MPC (FPC) in the specified slot number only on the specified member router in the Virtual Chassis.

For example, if you issue the **set chassis member 0 fpc slot 1 power off** statement, only the MPC installed in slot 1 of member ID 0 in the Virtual Chassis is powered off.

- Statements included at the **[edit chassis fpc slot *slot-number*]** hierarchy level should be relocated to the **[edit chassis member *member-id* fpc slot *slot-number*]** hierarchy level to avoid errors.

BEST PRACTICE: To ensure that the statement you use to configure MPC chassis properties in a Virtual Chassis applies to the intended member router and MPC, always include the member *member-ID* option before the fpc keyword, where *member-id* is 0 or 1 for a two-member MX Series Virtual Chassis.

Virtual Chassis Control Protocol

An MX Series Virtual Chassis is managed by the Virtual Chassis Control Protocol (VCCP), which is a dedicated control protocol based on IS-IS. VCCP runs on the Virtual Chassis port interfaces and performs the following functions in the Virtual Chassis:

- Discovers and builds the Virtual Chassis topology
- Runs the mastership election algorithm to determine the Virtual Chassis master router
- Establishes the interchassis routing table to route traffic within the Virtual Chassis

Like IS-IS, VCCP exchanges link-state PDUs for each member router to construct a shortest path first (SPF) topology and to determine each member router's role (master or backup) in the Virtual Chassis. Because VCCP supports only point-to-point connections, no more than two member routers can be connected on any given Virtual Chassis port interface.

Member IDs, Roles, and Serial Numbers

To configure an MX Series Virtual Chassis, you must create a preprovisioned configuration that provides the following required information for each member router:

- Member ID—A numeric value (**0** or **1**) that identifies the member router in a Virtual Chassis configuration.
- Role—The role to be performed by each member router in the Virtual Chassis. In a two-member MX Series Virtual Chassis, you must assign both member routers the **routing-engine** role, which enables either router to function as the master router or backup router of the Virtual Chassis.
- Serial number—The chassis serial number of each member router in the Virtual Chassis. To obtain the router's serial number, find the label affixed to the side of the MX Series chassis, or issue the **show chassis hardware** command on the router to display the serial number in the command output.

The preprovisioned configuration permanently associates the member ID and role with the member router's chassis serial number. When a new member router joins the Virtual

Chassis, the VCCP software compares the router's serial number against the values specified in the preprovisioned configuration. If the serial number of a joining router does not match any of the configured serial numbers, the VCCP software prevents that router from becoming a member of the Virtual Chassis.

Related Documentation

- Interchassis Redundancy and Virtual Chassis Overview on page 19
- Guidelines for Configuring Virtual Chassis Ports on page 134
- Global Roles and Local Roles in a Virtual Chassis on page 30
- Split Detection Behavior in a Virtual Chassis on page 38
- Virtual Chassis Slot Number Mapping for Use with SNMP on page 194
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Global Roles and Local Roles in a Virtual Chassis

In a Virtual Chassis configuration for MX Series 3D Universal Edge Routers or EX9200 switches, each of the two member devices and each of the two Routing Engines in each member device has a distinct role. A *global role* defines the function of each member device in the Virtual Chassis, and applies globally across the entire Virtual Chassis. A *local role* defines the function of each Routing Engine in the member device, and applies locally only to that member device.

Global roles change when you switch the Virtual Chassis mastership, and both global roles and local roles change when you switch the Routing Engine mastership in one of the member devices. In addition, the **line-card** global role, though not supported in a preprovisioned configuration for a two-member MX Series or EX9200 Virtual Chassis, applies in the context of split detection behavior.

This topic describes the global roles and local roles in a MX Series or EX9200 Virtual Chassis so you can better understand how the Virtual Chassis behaves during a global mastership switch, a local Routing Engine switchover, or when split detection is enabled.

- Role Name Format on page 30
- Global Role and Local Role Descriptions on page 31

Role Name Format

The global and local role names in an MX Series or EX9200 Virtual Chassis use the following format:

VC-*GlobalRole*<*LocalRole*>

where:

- ***GlobalRole*** applies to the global function of the member device for the entire Virtual Chassis, and can be one of the following:
  - **M**—Virtual Chassis master device, also referred to as the protocol master.

  - **B**—Virtual Chassis backup device, also referred to as the protocol backup.
  - **L**—Virtual Chassis line-card device. The **line-card** role is not supported in the preprovisioned configuration for a two-member Virtual Chassis. The **line-card** role applies only in the context of split detection behavior.

- ***LocalRole*** (optional) applies to the function of the Routing Engine in the local member device, and can be one of the following:
  - **m**—Master Routing Engine
  - **s**—Standby Routing Engine

Global Role and Local Role Descriptions

Table 5 on page 31 describes the global roles and local roles in an MX Series or EX9200 Virtual Chassis.

Table 5: Global Roles and Local Roles

Virtual Chassis Role	Type of Role	Description
VC-M	Global	Master device for the Virtual Chassis
VC-B	Global	Backup device for the Virtual Chassis
VC-L	Global	Line-card device for the Virtual Chassis NOTE: The **line-card** role is not supported in the preprovisioned configuration for a two-member MX Series or EX9200 Virtual Chassis. The **line-card** role applies only in the context of split detection behavior.
VC-Mm	Local	Master Routing Engine in the Virtual Chassis master device
VC-Ms	Local	Standby Routing Engine in the Virtual Chassis master device
VC-Bm	Local	Master Routing Engine in the Virtual Chassis backup device
VC-Bs	Local	Standby Routing Engine in the Virtual Chassis backup device
VC-Lm	Local	Master Routing Engine in the Virtual Chassis line-card device NOTE: The **line-card** role is not supported in the preprovisioned configuration for a two-member MX Series or EX9200 Virtual Chassis. The **line-card** role applies only in the context of split detection behavior.

Table 5: Global Roles and Local Roles *(continued)*

Virtual Chassis Role	Type of Role	Description
VC-Ls	Local	Standby Routing Engine in the Virtual Chassis line-card device NOTE: The **line-card** role is not supported in the preprovisioned configuration for a two-member MX Series or EX9200 Virtual Chassis. The **line-card** role applies only in the context of split detection behavior.

Related Documentation

- Virtual Chassis Components Overview on page 24
- *Understanding EX9200 Virtual Chassis*
- Mastership Election in a Virtual Chassis on page 32
- Switching the Global Master and Backup Roles in a Virtual Chassis Configuration on page 95
- Disabling Split Detection in a Virtual Chassis Configuration on page 98

Mastership Election in a Virtual Chassis

In a two-member MX Series or EX9200 Virtual Chassis, either member device can be elected as the master device (also known as the protocol master, or VC-M) of the Virtual Chassis. The first member device to join the Virtual Chassis becomes the initial master device by default. After the Virtual Chassis is formed with both member devices, the Virtual Chassis Control Protocol (VCCP) software runs a mastership election algorithm to elect the master device for the Virtual Chassis configuration.

If the master device in a Virtual Chassis fails, the backup device (also known as the protocol backup, or VC-B) takes over mastership of the Virtual Chassis. You can also switch the global roles of the master device and backup device in a Virtual Chassis by issuing the **request virtual-chassis routing-engine master switch** command.

NOTE: **You cannot configure mastership election for an MX Series or EX9200 Virtual Chassis in the current release.**

The VCCP software uses the following algorithm to elect the master device for an EX9200 or MX Series Virtual Chassis:

1. Choose the member device that has the highest value for the internal mastership election flag.

   The mastership election algorithm uses an internal flag that keeps track of the member state for the purpose of electing the Virtual Chassis master device. In most cases, VCCP elects the member device with the higher flag value over the member device with the lower flag value as the protocol master.

To display the mastership election flag value, issue the show virtual-chassis protocol database **extensive** command. The flag value used for mastership election appears in the **TLVs** field of the command output, as shown in the following example:

```
{master:member1-re0}

user@host> show virtual-chassis protocol database member 0 extensive
 ...
  TLVs:
    Node Info: Member ID: 1, VC ID: 5a6a.e747.8511, Flags: 3, Priority: 129
       System ID: 001d.b510.0800, Device ID: 1
 ...
```

2. Choose the member device with the highest mastership priority value.

   The mastership priority value is assigned to the member device by the VCCP software, and is not configurable in the current release. The mastership priority value can be one of the following:

   - **129**—The **routing-engine** role is assigned to the member device.
   - **128**—No role is assigned to the member device.
   - **0**—The **line-card** role is assigned to the member device (not supported in the current release).

   To display the mastership priority value for the member devices in the Virtual Chassis, issue the show virtual-chassis status command.

3. Choose the member device that is active in the Virtual Chassis.
4. Choose the member device that belongs to the Virtual Chassis with the largest number of members.

   NOTE: This criterion is not used in the current release because all EX9200 and MX Series Virtual Chassis configurations have two member devices.

5. Choose the member device that is the accepted (elected) protocol master of the Virtual Chassis.
6. Choose the member device that is the current protocol master (VC-M) of the same Virtual Chassis.
7. Choose the member device that is the current protocol backup (VC-B) of the same Virtual Chassis.
8. Choose the member device that has been part of the Virtual Chassis configuration for the longest period of time.
9. Choose the member device that was the previous protocol master of the same Virtual Chassis.
10. Choose the member device with the lowest media access control (MAC) address.

Related Documentation

- Virtual Chassis Components Overview on page 24
- Global Roles and Local Roles in a Virtual Chassis on page 30
- Switching the Global Master and Backup Roles in a Virtual Chassis Configuration on page 95

Switchover Behavior in an MX Series Virtual Chassis

When an active or primary hardware or software component fails or is temporarily shut down, you can manually initiate a *switchover* to a backup component that takes over the functions of the unavailable primary component. You can initiate two types of switchovers in a Virtual Chassis configuration for MX Series 3D Universal Edge Routers:

- Global switchover—Changes the mastership in an MX Series Virtual Chassis by switching the global roles of the master router and backup router in the Virtual Chassis configuration.
- Local switchover—Toggles the local mastership of the dual Routing Engines in a member router of the Virtual Chassis.

During a switchover, the roles assigned to the member routers and Routing Engines in a Virtual Chassis configuration change. This topic describes the role transitions that occur so you can better understand how an MX Series Virtual Chassis behaves during a global or local switchover. The topic also describes how you can determine whether the member routers are ready for a global graceful Routing Engine switchover (GRES) operation from a database synchronization perspective.

- Virtual Chassis Role Transitions During a Global Switchover on page 34
- Virtual Chassis Role Transitions During a Local Switchover on page 35
- Virtual Chassis Role Transitions During Virtual Chassis Formation on page 36
- GRES Readiness in a Virtual Chassis Configuration on page 37

Virtual Chassis Role Transitions During a Global Switchover

To change the mastership in an MX Series Virtual Chassis and cause a global switchover, you issue the **request virtual-chassis routing-engine master switch** command from the master Routing Engine in the Virtual Chassis master router (VC-Mm).

After you issue the **request virtual-chassis routing-engine master switch** command, the current Virtual Chassis master router (VC-M) and the current Virtual Chassis backup router (VC-B) switch roles. The former VC-M becomes the new VC-B, and the former VC-B becomes the new VC-M. After the VC-M and VC-B switch roles, the master Routing Engine on the new VC-B (VC-Bm) reboots, causing the role transitions listed in Table 6 on page 35.

Table 6: Virtual Chassis Role Transitions During Global Switchover

Virtual Chassis Role *Before* Global Switchover	Virtual Chassis Role *After* Global Switchover
Virtual Chassis master router (VC-M)	Virtual Chassis backup router (VC-B)
Virtual Chassis backup router (VC-B)	Virtual Chassis master router (VC-M)
Master Routing Engine in the Virtual Chassis master router (VC-Mm)	Standby Routing Engine in the Virtual Chassis backup router (VC-Bs)
Standby Routing Engine in the Virtual Chassis master router (VC-Ms)	Master Routing Engine in the Virtual Chassis backup router (VC-Bm)
Master Routing Engine in the Virtual Chassis backup router (VC-Bm)	Master Routing Engine in the Virtual Chassis master router (VC-Mm)
Standby Routing Engine in the Virtual Chassis backup router (VC-Bs)	Standby Routing Engine in the Virtual Chassis master router (VC-Ms)

The local roles (**master** and **standby**, or **m** and **s**) of the Routing Engines in the Virtual Chassis master router change after a global switchover, but the local roles of the Routing Engines in the Virtual Chassis backup router do not change. For example, as shown in Table 6 on page 35, the master Routing Engine in the Virtual Chassis master router (VC-Mm) becomes the standby Routing Engine in the Virtual Chassis backup router (VC-Bs) after the global switchover. By contrast, the master Routing Engine in the Virtual Chassis backup router (VC-Bm) remains the master Routing Engine in the Virtual Chassis master router (VC-Mm) after the global switchover.

Virtual Chassis Role Transitions During a Local Switchover

To ensure redundancy in a two-member Virtual Chassis configuration, each of the two member routers must be configured with dual Routing Engines. To toggle local mastership between the master Routing Engine and the standby Routing Engine in the member router, you issue the **request chassis routing-engine master switch** command from *either* the master Routing Engine in the Virtual Chassis master router (VC-Mm) or from the master Routing Engine in the Virtual Chassis backup router (VC-Bm).

Table 7 on page 35 shows the role transitions caused by a local switchover when you issue the **request chassis routing-engine master switch** command from the VC-Mm.

Table 7: Virtual Chassis Role Transitions During Local Switchover Performed from VC-Mm

Virtual Chassis Role *Before* Local Switchover	Virtual Chassis Role *After* Local Switchover
Master Routing Engine in the Virtual Chassis master router (VC-Mm)	Standby Routing Engine in the Virtual Chassis backup router (VC-Bs)
Standby Routing Engine in the Virtual Chassis master router (VC-Ms)	Master Routing Engine in the Virtual Chassis backup router (VC-Bm)

Table 7: Virtual Chassis Role Transitions During Local Switchover Performed from VC-Mm *(continued)*

Virtual Chassis Role *Before* Local Switchover	Virtual Chassis Role *After* Local Switchover
Master Routing Engine in the Virtual Chassis backup router (VC-Bm)	Master Routing Engine in the Virtual Chassis master router (VC-Mm)
Standby Routing Engine in the Virtual Chassis backup router (VC-Bs)	Standby Routing Engine in the Virtual Chassis master router (VC-Ms)

Table 8 on page 36 shows the role transitions caused by a local switchover when you issue the **request chassis routing-engine master switch** command from the VC-Bm.

Table 8: Virtual Chassis Role Transitions During Local Switchover Performed from VC-Bm

Virtual Chassis Role *Before* Local Switchover	Virtual Chassis Role *After* Local Switchover
Master Routing Engine in the Virtual Chassis backup router (VC-Bm)	Standby Routing Engine in the Virtual Chassis backup router (VC-Bs)
Standby Routing Engine in the Virtual Chassis backup router (VC-Bs)	Master Routing Engine in the Virtual Chassis backup router (VC-Bm)
Master Routing Engine in the Virtual Chassis master router (VC-Mm)	Master Routing Engine in the Virtual Chassis master router (VC-Mm)
Standby Routing Engine in the Virtual Chassis master router (VC-Ms)	Standby Routing Engine in the Virtual Chassis master router (VC-Ms)

When you perform a local switchover, the master (m) and standby (s) local roles of the Routing Engines in each member router change only in the member router from which you issue the **request chassis routing-engine master switch** command. For example, when you issue a local switchover from the VC-Mm, as shown in Table 7 on page 35, the local roles change on the VC-M but remain the same on the VC-B. Conversely, when you issue a local switchover from the VC-Bm, as shown in Table 8 on page 36, the local roles change on the VC-B but remain the same on the VC-M.

A local switchover performed from the VC-Mm also changes the global roles of the member routers, as shown in Table 7 on page 35. By contrast, a local switchover performed from the VC-Bm changes only the local roles of the Routing Engines, as shown in Table 8 on page 36.

Virtual Chassis Role Transitions During Virtual Chassis Formation

In the rare case when the virtual chassis has "split" (that is, lost connectivity), each member may take the Virtual Chassis master router (VC-M) role, resulting in two VC-M chassis. When Virtual Chassis connectivity is restored, an election process assigns the Virtual Chassis master (VC-M) role to one member and the Virtual Chassis backup (VC-B)

role to the other member. As of Junos OS Release 15.1, in the same manner as the global GRES behavior, the newly elected VC-B member causes its local master Routing Engine to reboot after passing local mastership to its local standby Routing Engine. This is an intentional action which allows the VC-B chassis to become GRES-ready more quickly.

NOTE: **Rebooting both Routing Engines in the VC-M chassis may not result in a graceful switchover to the VC-B chassis, and is not recommended.**

GRES Readiness in a Virtual Chassis Configuration

Depending on the configuration, a variable amount of time is required before a router is ready to perform a graceful Routing Engine switchover (GRES). Attempting a GRES operation before the router is ready can cause system errors and unexpected behavior. To determine whether the member routers in an MX Series Virtual Chassis configuration are ready for a GRES operation from a database synchronization perspective, you can issue the **request virtual-chassis routing-engine master switch check** command from the Virtual Chassis master router (VC-Mm) before you initiate the GRES operation.

The **request virtual-chassis routing-engine master switch check** command checks various system and database components on the member routers to determine whether they are ready for GRES, but does not initiate the global GRES operation itself. The readiness check includes ensuring that a system timer, which expires after 300 seconds, completes before the global GRES operation begins.

Using the **request virtual-chassis routing-engine master switch check** command before you initiate the GRES operation ensures that the subscriber management and kernel databases on both member routers in an MX Series or Virtual Chassis are synchronized and ready for the GRES operation.

Related Documentation

- Switching the Global Master and Backup Roles in a Virtual Chassis Configuration on page 95
- Determining GRES Readiness in a Virtual Chassis Configuration on page 185
- Virtual Chassis Components Overview on page 24
- Global Roles and Local Roles in a Virtual Chassis on page 30
- Mastership Election in a Virtual Chassis on page 32
- *Understanding Graceful Routing Engine Switchover*

Split Detection Behavior in a Virtual Chassis

If there is a disruption to a Virtual Chassis configuration for MX Series 3D Universal Edge Routers or EX9200 Switches due to the failure of a member router or switch or one or more Virtual Chassis port interfaces, the resulting connectivity loss can cause a split in the Virtual Chassis configuration. *Split detection* identifies the split and can minimize further network disruption.

This topic covers:

- How Split Detection Works in a Virtual Chassis on page 38
- Effect of Split Detection on Virtual Chassis Failure Scenarios on page 38

How Split Detection Works in a Virtual Chassis

Split detection is enabled by default in an EX9200 or MX Series Virtual Chassis. Optionally, you can disable split detection by including the **no-split-detection** statement at the **[edit virtual-chassis]** hierarchy level. Disabling split detection can be useful in certain Virtual Chassis configurations.

For example, if the backup router or switch fails in a two-member Virtual Chassis configuration and split detection is enabled (the default behavior), the master router or switch takes a **line-card** role, and the line cards (FPCs) that do not host Virtual Chassis ports go offline. This state effectively halts routing and disables the Virtual Chassis configuration. By contrast, if the backup router or switch fails in a two-member Virtual Chassis configuration and split detection is disabled, the master router or switch retains mastership and maintains all of the Virtual Chassis ports, effectively resulting in a single-member Virtual Chassis consisting of only the master router or switch.

BEST PRACTICE: We recommend that you disable split detection for a two-member Virtual Chassis configuration if you think the backup router or switch is more likely to fail than the Virtual Chassis port interfaces to the backup router or switch. Configuring redundant Virtual Chassis ports on different line cards in each member router or switch reduces the likelihood that all Virtual Chassis port interfaces to the backup router or switch can fail.

Effect of Split Detection on Virtual Chassis Failure Scenarios

The behavior of a Virtual Chassis during certain failure scenarios depends on whether split detection is enabled or disabled. Table 9 on page 39 describes the effect of the split detection setting on common failure scenarios in a two-member MX Series Virtual Chassis.

Table 9: Effect of Split Detection on Common Virtual Chassis Failure Scenarios

Type of Failure	Split Detection Setting	Results
Virtual Chassis port interfaces go down	Enabled	• VC-B takes VC-M role. • Previous VC-M takes **line-card** (VC-L) role. The **line-card** role isolates the router or switch and removes it from the Virtual Chassis until connectivity is restored. • Result is a single-member Virtual Chassis consisting of only a single VC-M. The VC-M continues to maintain subscriber state information and route traffic. When Virtual Chassis port interfaces are reconnected: • VC-M retains VC-M role. • VC-L takes VC-B role. • Subscribers are not affected.
Virtual Chassis port interfaces go down	Disabled	When Virtual Chassis port interfaces are disconnected: • VC-M retains VC-M role, and VC-B also takes VC-M role. The result is a Virtual Chassis with two VC-M routers or switches, each of which maintains subscriber state information. • Initially, both VC-M routers or switches have a complete list of subscribers. Because the two routers or switches have the same configuration, the effect on subscribers, traffic patterns, behavior of external applications, and subscriber login and logout operations is unpredictable while the Virtual Chassis port interfaces are disconnected. When Virtual Chassis port interfaces are reconnected: • Original VC-M before the disconnection resumes VC-M role, and original VC-B before the disconnection resumes VC-B role. • Subscribers on the VC-M are preserved. • Subscribers on the VC-B are purged. • The subscribers preserved on the VC-M are unaffected, and all remaining subscribers are able to log back in to the router or switch.

Table 9: Effect of Split Detection on Common Virtual Chassis Failure Scenarios *(continued)*

Type of Failure	Split Detection Setting	Results
Virtual Chassis backup router or switch (VC-B) goes down	Enabled	• VC-M takes **line-card** (VC-L) role, which causes all line cards (FPCs) that do not host Virtual Chassis ports to go offline. • Previous VC-B is out of service. • The **line-card** role isolates the master router or switch and removes it from the Virtual Chassis until connectivity is restored. As a result, the Virtual Chassis is left without a master router or switch, which halts interchassis routing and effectively disables the Virtual Chassis configuration. When the failed router or switch is brought back into service: • The mastership election algorithm is run to determine whether the router or switch takes a VC-M or VC-B role. The Virtual Chassis then becomes operational. • All subscribers can log back in to the router or switch. • Previous subscriber state information is not preserved.
Virtual Chassis backup router or switch (VC-B) goes down	Disabled	• VC-M retains VC-M role and maintains all Virtual Chassis ports. • Previous VC-B is out of service. • Result is a single-member Virtual Chassis consisting of only a single VC-M. The VC-M continues to maintain subscriber state information and route traffic.
Virtual Chassis master router or switch (VC-M) goes down	Split detection setting has no effect on behavior	• VC-B takes over VC-M role regardless of whether split detection is enabled or disabled. • Previous VC-M is out of service. • Result is a single-member Virtual Chassis consisting of only a single VC-M. The new VC-M continues to maintain subscriber state information and route traffic. When the original VC-M is brought back into service, or when the original VC-M is replaced with a new router or switch: • Original VC-M or its replacement takes VC-B role. • Subscribers are not affected.

Table 9: Effect of Split Detection on Common Virtual Chassis Failure Scenarios *(continued)*

Type of Failure	Split Detection Setting	Results
Active access link between the VC-M and the access node, such as a digital subscriber line access multiplexer (DSLAM), goes down	Split detection setting has no effect on behavior	• Previous standby access link becomes the active access link between the VC-B and the access node. • Traffic is routed through the new active access link. • The VC-M continues to maintain subscriber state information and route traffic.

Related Documentation

- Virtual Chassis Components Overview on page 24
- Global Roles and Local Roles in a Virtual Chassis on page 30
- Mastership Election in a Virtual Chassis on page 32
- Switchover Behavior in an MX Series Virtual Chassis on page 34
- Disabling Split Detection in a Virtual Chassis Configuration on page 98

Virtual Chassis Heartbeat Connection Overview

Starting in Junos OS Release 14.1, you can configure an IP-based, bidirectional "heartbeat" packet connection between the master router and backup router in an MX Series Virtual Chassis. You can use this heartbeat connection to determine the health and availability of member routers in the Virtual Chassis. The member routers forming this heartbeat connection exchange *heartbeat packets* that provide critical information about the availability and health of each member router. During a disruption or split in the Virtual Chassis configuration, the heartbeat connection prevents the member routers from changing mastership roles unnecessarily. Without the heartbeat connection, a change in mastership roles in such a situation can produce undesirable results, such as having two Virtual Chassis master routers or no Virtual Chassis master router.

- Benefits of Configuring a Virtual Chassis Heartbeat Connection on page 41
- Configuration Requirements for the Heartbeat Connection on page 42
- How the Heartbeat Connection Works on page 44
- Heartbeat Connection and Virtual Chassis Failure Conditions on page 45
- Heartbeat Connection Compared to Split Detection on page 45

Benefits of Configuring a Virtual Chassis Heartbeat Connection

Configuring a Virtual Chassis heartbeat connection provides the following benefits for an MX Series Virtual Chassis:

- Improved resiliency during failure scenarios

Configuring the heartbeat connection improves resiliency of the Virtual Chassis in the event of an adjacency disruption or split caused by a failure of the Virtual Chassis port interfaces, or when one of the member routers goes out of service. If the heartbeat connection detects that the Virtual Chassis master router (VC-M) is still operating and able to respond during a split, the software maintains mastership on the existing VC-M, isolates the Virtual Chassis backup router (VC-B) until the Virtual Chassis recovers, and resumes the backup role on the VC-B when the Virtual Chassis forms again. As a result, the heartbeat connection prevents the member routers from unnecessarily changing mastership roles, which consumes system resources and causes unexpected and undesirable results.

When the VC-B is isolated during a disruption, the software immediately restarts all line cards and powers off all network ports until the disruption is resolved and the Virtual Chassis forms again. This behavior supports network applications with external equipment that requires a physical link-down condition to switch the traffic paths to other connections.

- Enhanced mastership election process

  The Virtual Chassis Control Protocol (VCCP) controls mastership election in a Virtual Chassis. When you configure the heartbeat connection in an MX Series Virtual Chassis, the VCCP software assesses the health information collected from the heartbeat connection to help determine which member router should become the global master (VC-M) in the event of an adjacency disruption or split. When the heartbeat connection detects that the peer member router is responsive, the VCCP software suppresses unnecessary changes in mastership roles.

  By contrast, when the heartbeat connection is *not* configured, the VCCP software does not have this additional health information when determining the appropriate mastership roles after a disruption or split.

- Ability to easily view and clear statistics related to the heartbeat connection

  Operational commands for the Virtual Chassis enable you to display the status of the heartbeat connection, review detailed statistics and latency measurements related to the heartbeat connection, and clear heartbeat-related statistics counters and timestamp fields for one or both member routers.

Configuration Requirements for the Heartbeat Connection

To establish a heartbeat connection for an MX Series Virtual Chassis, you must configure a secure and reliable route between the master router and backup router for the exchange of TCP/IP heartbeat packets. Specifically, you must ensure that the master Routing Engine in the Virtual Chassis backup router (VC-Bm) can make a TCP/IP connection to the **master-only** IP address of the master Routing Engine in the Virtual Chassis master router (VC-Mm).

The following additional requirements apply when you configure the heartbeat connection:

- Configure the heartbeat connection only between Virtual Chassis member routers eligible to become the Virtual Chassis master router, also known as the *protocol master* or *global master*.

  In a two-member MX Series Virtual Chassis configuration, you assign the **routing-engine** role to each router as part of the preprovisioned configuration. The **routing-engine** role enables the router to function either as the master router or backup router of the Virtual Chassis as needed. As a result, you can configure the heartbeat connection between both member routers in a two-member MX Series Virtual Chassis configuration.

- Use the router's Ethernet management interface (**fxp0**) as the heartbeat path.

  The management interface is generally available earlier than the line card interfaces, and is typically connected to a more secure network than the other interfaces.

- Configure a **master-only** IP address for the **fxp0** management interface to ensure consistent access to the VC-Mm, regardless of which Routing Engine is currently active.

  The **master-only** address is active only on the management interface for the VC-Mm. During a switchover, the **master-only** address moves to the new Routing Engine currently functioning as the VC-Mm.

- Ensure TCP connectivity between the VC-Mm and VC-Bm member routers

  The Virtual Chassis heartbeat connection opens a proprietary TCP port numbered 33087 on the VC-Mm to listen for heartbeat messages. If your network design includes firewalls or filters, make sure the network allows traffic between TCP port 33087 on the VC-Mm and the dynamically allocated TCP port on the VC-Bm.

- When using a heartbeat connection, do not configure the **no-split-detection** statement as part of the preprovisioned Virtual Chassis configuration.

  The **no-split-detection** statement suppresses any action when a split is detected in the Virtual Chassis. Using the **no-split-detection** statement is prohibited when you configure a heartbeat connection, and the software prevents you from configuring both the **no-split-detection** and **heartbeat-address** statements at the same time. If you attempt to do so, the software displays an error message and causes the commit operation to fail.

In a two-member MX Series Virtual Chassis, you can configure a heartbeat connection with both member routers in the same subnet, or with each member router in a different subnet. Table 10 on page 44 summarizes the important differences between the configuration procedures for member routers in the same subnet and member routers in different subnets.

Table 10: Comparison of Heartbeat Connection Configuration Tasks for Member Routers in Same Subnet and Member Routers in Different Subnets

Task	Heartbeat Connection for Member Routers in *Same Subnet*	Heartbeat Connection for Member Routers in *Different Subnets*
Configure the **master-only** IP address for **fxp0** management interface.	Configure the same **fxp0 master-only** IP address for all four member Routing Engines.	Configure two different **master-only** IP addresses for the **fxp0** management interface: one address for the subnet in which the Virtual Chassis master router resides, and one for the subnet in which the backup router resides.
Configure a network path for the heartbeat connection.	Provide a path for the member routers to reach each other by means of a TCP/IP connection. For example, in a Virtual Chassis with member routers in the same subnet, you can use the router's default gateway. Alternatively, you can create a global static route as described in "Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet" on page 196.	Provide a path for the member routers to reach each other by means of a TCP/IP connection. In a Virtual Chassis with member routers in different subnets, you must ensure that both member routers can reach each other's network. For example, you can create static routes to both subnets on each member Routing Engine, as described in "Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets" on page 207.
Configure the heartbeat address to establish the heartbeat connection.	Configure a single (global) **master-only** IP address for the **fxp0** management interface as the heartbeat address to establish the connection.	Configure a heartbeat address for each member Routing Engine to cross-connect to the **master-only** IP address for the corresponding Routing Engine in the other subnet. For example, assume that **member0-re0** and **member0-re1** reside in subnet 10.4.0.0, and **member1-re0** and **member1-re1** reside in subnet 10.5.0.0. In this configuration, you would set the heartbeat address for **member0-re0** to the **master-only** IP address for **member1-re0** to cross-connect **member0-re0** and **member1-re0**. You would cross-connect **member0-re1** and **member1-re1** in a similar manner.

How the Heartbeat Connection Works

When the Virtual Chassis is operating properly, the heartbeat connection periodically sends heartbeat packets over the TCP/IP path between the master Routing Engine in the Virtual Chassis master router and the master Routing Engine in the Virtual Chassis backup router.

When an adjacency disruption or split is detected in the Virtual Chassis, each member router sends a final heartbeat message to determine whether the other member is able to respond, and stops sending additional periodic messages until the Virtual Chassis forms again. The other member must respond to the heartbeat message within the default heartbeat timeout period (2 seconds), or within a configured heartbeat timeout period in the range 1 through 60 seconds. To determine the time period that elapses in your network between transmission of a heartbeat request message and receipt of a heartbeat response message, you can issue the **show virtual-chassis heartbeat detail**

command to view the number of seconds reported in the **Maximum latency** and **Minimum latency** fields.

BEST PRACTICE: If your network is congested or has a round-trip latency that exceeds 2 seconds, we recommend that you increase the value of the heartbeat timeout period to account for this delay during a Virtual Chassis adjacency disruption or split.

Heartbeat Connection and Virtual Chassis Failure Conditions

Configuring the heartbeat connection prevents unnecessary mastership role changes between the Virtual Chassis member routers when an adjacency disruption or split occurs. Table 11 on page 45 describes the effects on mastership for common failure conditions when you enable the heartbeat connection in a two-member MX Series Virtual Chassis.

Table 11: Effect of Heartbeat Connection on Common Virtual Chassis Failure Conditions

Failure Condition	Result on Virtual Chassis Master Router (VC-M)	Result on Virtual Chassis Backup Router (VC-B)
Virtual Chassis port interfaces go down.	Retains VC-M role.	If the VC-M is in service but the Virtual Chassis port interfaces are down, the VC-B goes offline after the heartbeat timeout period expires because the Routing Engine state is invalid.
VC-M chassis fails.	Goes out of service.	Becomes VC-M.
VC-B chassis fails.	Retains VC-M role.	Goes out of service.
Heartbeat connection fails.	Retains VC-M role.	Retains VC-B role.

In all cases except when the VC-M chassis fails, mastership of the Virtual Chassis is maintained on the existing VC-Mm if the heartbeat connection detects that the VC-M is still operating and able to respond during a split. Preventing an unnecessary role change minimizes the system load caused by a protocol mastership switch, and reduces the liklihood of unpredictable results.

Heartbeat Connection Compared to Split Detection

In certain Virtual Chassis failure conditions, the split detection setting (enabled by default, or explicitly disabled) can cause unpredictable and undesirable results such as a Virtual Chassis with two master routers, or a Virtual Chassis with no master router. Table 12 on page 46 compares the effects of split detection and the heartbeat connection for two common failure conditions: failure of the Virtual Chassis port interfaces and failure of the VC-B chassis.

Table 12: Comparison of Heartbeat Connection and Split Detection for Virtual Chassis Failure Conditions

Failure Condition	Results with Heartbeat Connection	Results with Split Detection
Virtual Chassis port interfaces go down.	• VC-M chassis retains VC-M role. • If the VC-M chassis is in service but the Virtual Chassis port interfaces are down, the VC-B chassis goes offline after the heartbeat timeout period expires because the Routing Engine state is invalid.	When split detection is disabled: • VC-M chassis retains VC-M role. • VC-B chassis also takes VC-M role. • Virtual Chassis has two master routers, each of which maintains subscriber state information. The effect on subscribers, traffic patterns, behavior of external applications, and subscriber login and logout operations is unpredictable while the Virtual Chassis port interfaces are disconnected.
VC-B chassis fails.	• VC-M chassis retains VC-M role. • VC-B chassis is out of service.	When split detection is enabled: • VC-M chassis takes line-card (VC-L) role, which isolates and removes it from the Virtual Chassis until connectivity is restored. • VC-B chassis is out of service. • Virtual Chassis does not have a master router. This state halts interchassis routing and effectively disables the Virtual Chassis configuration.

BEST PRACTICE: We recommend that you use the heartbeat connection instead of the split detection feature in an MX Series Virtual Chassis to avoid unnecessary mastership role changes during an adjacency disruption or split, and to provide additional member health information for the mastership election process.

Release History Table

Release	Description
14.1	Starting in Junos OS Release 14.1, you can configure an IP-based, bidirectional “heartbeat” packet connection between the master router and backup router in an MX Series Virtual Chassis.

Related Documentation

- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet on page 196
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets on page 207
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

- Global Roles and Local Roles in a Virtual Chassis on page 30
- Split Detection Behavior in a Virtual Chassis on page 38
- *Configuring a Consistent Management IP Address*

Command Forwarding in a Virtual Chassis

You can run some CLI commands on all member routers, on the local member router, or on a specific member router in an MX Series Virtual Chassis configuration. This feature is referred to as *command forwarding*. With command forwarding, the router sends the command to the specified member router or routers, and displays the results as if the command were processed on the local router.

For example, to collect information about your system prior to contacting Juniper Networks Technical Assistance Center (JTAC), use the command **request support information all-members** to gather data for all the member routers. If you want to gather this data only for a particular member router, use the command **request support information member *member-id***.

Table 13 on page 47 describes the commands that you can run on all (both) member routers (with the **all-members** option), on the local member router (with the **local** option), or on a specific member router (with the **member *member-id*** option) in an MX Series Virtual Chassis configuration, where *member-id* is 0 or 1.

Table 13: Commands Available for Command Forwarding in an MX Series Virtual Chassis

Command	Purpose	all-members	local	member *member-id*
request chassis fpc	Control the operation of the Flexible PIC Concentrator (FPC).	Change FPC status of all members of the Virtual Chassis configuration.	(Default) Change FPC status of the local member of the Virtual Chassis.	Change FPC status of the specified member of the Virtual Chassis.
request chassis fpm resync	Resynchronize the craft interface status.	Resynchronize the craft interface status on all members of the Virtual Chassis configuration.	(Default) Resynchronize the craft interface status on the local member of the Virtual Chassis.	Resynchronize the craft interface status on the specified member of the Virtual Chassis.
request chassis pic	Change the PIC status of the specified member router.	—	—	Change the PIC status on the specified member of the Virtual Chassis.
request chassis routing-engine master	Control which Routing Engine is the master for a router with dual Routing Engines.	Control Routing Engine mastership on the Routing Engines in all member routers of the Virtual Chassis configuration.	(Default) Control Routing Engine mastership on the Routing Engines in the local Virtual Chassis configuration.	Control Routing Engine mastership on the Routing Engines of the specified member in the Virtual Chassis configuration.

Table 13: Commands Available for Command Forwarding in an MX Series Virtual Chassis *(continued)*

Command	Purpose	all-members	local	member *member-id*
request chassis sfb	(MX2010 and MX2020 routers only) Control the operation of the Switch Fabric Board (SFB).	Control the operation of the SFB in all members of the Virtual Chassis configuration.	(Default) Control the operation of the SFB in the local member of the Virtual Chassis.	Control the operation of the SFB in the specified member of the Virtual Chassis.
request chassis spmb restart	(MX2010 and MX2020 routers only) Restart the specified Switch Processor Mezzanine Board (SPMB) on the Control Board (CB).	Restart the SPMB on the CB in all members of the Virtual Chassis configuration.	(Default) Restart the SPMB on the CB in the local member of the Virtual Chassis.	Restart the SPMB on the CB in the specified member of the Virtual Chassis.
request routing-engine login	Specify a **tty** connection for login for a router with two Routing Engines.	Log in to all members of the Virtual Chassis configuration.	(Default) Log in to the local member of the Virtual Chassis.	Log in to the specified member of the Virtual Chassis.
request support information	Display information about the system.	(Default) Display system information for all members of the Virtual Chassis configuration.	Display system information for the local member of the Virtual Chassis.	Display system information for the specified member of the Virtual Chassis.
request system halt	Stop the router.	(Default) Halt all members of the Virtual Chassis configuration.	Halt the local member of the Virtual Chassis.	Halt the specified member of the Virtual Chassis.
request system partition abort	Terminate a previously scheduled storage media partition operation.	(Default) Abort a previously scheduled storage media partition operation for all members of the Virtual Chassis configuration.	Abort a previously scheduled storage media partition operation for the local member of the Virtual Chassis.	Abort a previously scheduled storage media partition operation for the specified member of the Virtual Chassis member.
request system partition hard-disk	Set up the hard disk for partitioning.	(Default) Schedule a partition of the hard disk for all members of the Virtual Chassis configuration.	Schedule a partition of the hard disk for the local member of the Virtual Chassis.	Schedule a partition of the hard disk for the specified member of the Virtual Chassis.
request system power-off	Power off the software.	(Default) Power off all members of the Virtual Chassis configuration.	Power off the local member of the Virtual Chassis.	Power off the specified member of the Virtual Chassis.
request system reboot	Reboot the software.	(Default) Reboot the software on all members of the Virtual Chassis configuration.	Reboot the software on the local member of the Virtual Chassis.	Reboot the software on the specified member of the Virtual Chassis.

Table 13: Commands Available for Command Forwarding in an MX Series Virtual Chassis *(continued)*

Command	Purpose	all-members	local	member *member-id*
request system snapshot	Back up the currently running and active file system partitions on the router to standby partitions that are not running.	(Default) Archive data and executable areas for all members of the Virtual Chassis configuration.	Archive data and executable areas for the local member of the Virtual Chassis.	Archive data and executable areas for the specified member of the Virtual Chassis.
request system software add	Install a software package or bundle on the router.	(Default if no options specified) Install a software package on all members of the Virtual Chassis configuration.	—	Install a software package on the specified member of the Virtual Chassis.
request system software rollback	Revert to the software that was loaded at the last successful **request system software add** command.	(Default) Attempt to roll back to the previous set of packages on all members of the Virtual Chassis configuration.	Attempt to roll back to the previous set of packages on the local member of the Virtual Chassis.	Attempt to roll back to the previous set of packages on the specified member of the Virtual Chassis.
request system software validate	Validate candidate software against the current configuration of the router.	—	(Default if no options specified) Validate the software package on the local member of the Virtual Chassis.	Validate the software bundle or package on the specified member of the Virtual Chassis.
request system storage cleanup	Free storage space on the router or switch by rotating log files and proposing a list of files for deletion.	(Default) Delete files on all members of the Virtual Chassis configuration.	Delete files on the local member of the Virtual Chassis.	Delete files on the specified member of the Virtual Chassis.
restart	Restart a Junos OS process.	Restart the software process for all members of the Virtual Chassis configuration.	(Default) Restart the software process for the local member of the Virtual Chassis.	Restart the software process for a specified member of the Virtual Chassis.
show chassis adc	(MX2010 and MX2020 routers only) Display information about the adapter cards (ADCs).	(Default) Display information about the ADCs in all members of the Virtual Chassis configuration.	Display information about the ADCs in the local member of the Virtual Chassis.	Display information about the ADCs in the specified member of the Virtual Chassis.
show chassis alarms	Display information about the conditions that have been configured to trigger alarms.	(Default) Display information about alarm conditions for all the member routers of the Virtual Chassis configuration.	Display information about alarm conditions for the local member of the Virtual Chassis.	Display information about alarm conditions for the specified member of the Virtual Chassis.

Table 13: Commands Available for Command Forwarding in an MX Series Virtual Chassis *(continued)*

Command	Purpose	all-members	local	member *member-id*
show chassis craft-interface	View messages currently displayed on the craft interface.	(Default) Display information currently on the craft interface for all members of the Virtual Chassis configuration.	Display information currently on the craft interface for the local member of the Virtual Chassis.	Display information currently on the craft interface for the specified member of the Virtual Chassis.
show chassis environment	Display environmental information about the router or switch chassis, including the temperature and information about the fans, power supplies, and Routing Engine.	(Default) Display chassis environmental information for all the members of the Virtual Chassis configuration.	Display chassis environmental information for the local member of the Virtual Chassis.	Display chassis environmental information for the specified member of the Virtual Chassis.
show chassis environment adc	(MX2010 and MX2020 routers only) Display chassis environmental information about the adapter cards (ADCs).	(Default) Display chassis environmental information about the ADCs in all members of the Virtual Chassis configuration.	Display chassis environmental information about the ADCs in the local member of the Virtual Chassis.	Display chassis environmental information about the ADCs in the specified member of the Virtual Chassis.
show chassis environment cb	Display environmental information about the Control Boards (CBs).	(Default) Display environmental information about the CBs on all the members of the Virtual Chassis configuration.	Display environmental information about the CBs on the local member of the Virtual Chassis.	Display environmental information about the CBs on the specified member of the Virtual Chassis.
show chassis environment fan	(MX2010 and MX2020 routers only) Display environmental information about the fans and fan trays.	(Default) Display environmental information about the fans and fan trays in all members of the Virtual Chassis configuration.	Display environmental information about the fans and fan trays in the local member of the Virtual Chassis.	Display environmental information about the fans and fan trays in the specified member of the Virtual Chassis.
show chassis environment fpc	Display environmental information about Flexible PIC Concentrators (FPCs).	(Default) Display environmental information for the FPCs in all the members of the Virtual Chassis configuration.	Display environmental information for the FPCs in the local member of the Virtual Chassis.	Display environmental information for the FPCs in the specified member of the Virtual Chassis.
show chassis environment monitored	(MX2010 and MX2020 routers only) Display status information for monitored temperatures.	(Default) Display status information for monitored temperatures in all members of the Virtual Chassis configuration.	Display status information for monitored temperatures in the local member of the Virtual Chassis.	Display status information for monitored temperatures in the specified member of the Virtual Chassis.

Table 13: Commands Available for Command Forwarding in an MX Series Virtual Chassis *(continued)*

Command	Purpose	all-members	local	member *member-id*
show chassis environment pem	Display Power Entry Module (PEM) environmental status information.	(Default) Display environmental information about the PEMs in all the member routers of the Virtual Chassis configuration.	Display environmental information about the PEMs in the local member of the Virtual Chassis.	Display environmental information about the PEMs in the specified member of the Virtual Chassis.
show chassis environment psm	(MX2010 and MX2020 routers only) Display chassis environmental information about the power supply modules (PSMs).	(Default) Display chassis environmental information about the PSMs in all member routers of the Virtual Chassis configuration.	Display chassis environmental information about the PSMs in the local member of the Virtual Chassis.	Display chassis environmental information about the PSMs in the specified member of the Virtual Chassis.
show chassis environment routing-engine	Display Routing Engine environmental status information.	(Default) Display environmental information about the Routing Engines in all member routers in the Virtual Chassis configuration.	Display environmental information about the Routing Engines in the local member of the Virtual Chassis.	Display environmental information about the Routing Engines in the specified member of the Virtual Chassis.
show chassis environment sfb	(MX2010 and MX2020 routers only) Display chassis environmental information about the Switch Fabric Boards (SFBs).	(Default) Display chassis environmental information about the SFBs in all member routers in the Virtual Chassis configuration.	Display chassis environmental information about the SFBs in the local member of the Virtual Chassis.	Display chassis environmental information about the SFBs in the specified member of the Virtual Chassis.
show chassis ethernet-switch	Display information about the ports on the Control Board (CB) Ethernet switch.	(Default) Display information about the ports on the CB Ethernet switch on all the members of the Virtual Chassis configuration.	Display information about the ports on the CB Ethernet switch on the local member of the Virtual Chassis.	Display information about the ports on the CB Ethernet switch on the specified member of the Virtual Chassis.
show chassis fabric fpcs	Display the state of the electrical and optical switch fabric links between the Flexible PIC Concentrators (FPCs) and the Switch Interface Boards (SIBs).	(Default) Display the switching fabric link states for the FPCs in all members of the Virtual Chassis configuration.	Display the switching fabric link states for the FPCs in the local member of the Virtual Chassis.	Display the switching fabric link states for the FPCs in the specified member of the Virtual Chassis.
show chassis fabric map	Display the switching fabric map state.	(Default) Display the switching fabric map state for all the members of the Virtual Chassis configuration.	Display the switching fabric map state for the local member of the Virtual Chassis.	Display the switching fabric map state for the specified member of the Virtual Chassis.

Table 13: Commands Available for Command Forwarding in an MX Series Virtual Chassis *(continued)*

Command	Purpose	all-members	local	member *member-id*
show chassis fabric plane	Display the state of all fabric plane connections.	(Default) Display the state of all fabric plane connections on all members of the Virtual Chassis configuration.	Display the state of all fabric plane connections on the local member of the Virtual Chassis.	Display the state of all fabric plane connections on the specified member of the Virtual Chassis.
show chassis fabric plane-location	Display the Control Board (CB) location of each plane on both the master and backup Routing Engine.	(Default) Display the CB location of each fabric plane on the Routing Engines in all member routers in the Virtual Chassis configuration.	Display the CB location of each fabric plane on the Routing Engines in the local member of the Virtual Chassis.	Display the CB location of each fabric plane on the Routing Engines in the specified member in the Virtual Chassis configuration.
show chassis fan	Display information about the fan tray and fans.	(Default) Display information about the fan tray and fans for all members of the Virtual Chassis configuration.	Display information about the fan tray and fans for the local member of the Virtual Chassis.	Display information about the fan tray and fans for the specified member of the Virtual Chassis.
show chassis firmware	Display the version levels of the firmware running on the System Control Board (SCB), Switching and Forwarding Module (SFM), System and Switch Board (SSB), Forwarding Engine Board (FEB), Flexible PIC Concentrators (FPCs), and Routing Engines.	(Default) Display the version levels of the firmware running for all members of the Virtual Chassis configuration.	Display the version levels of the firmware running for the local member of the Virtual Chassis.	Display the version levels of the firmware running for the specified member of the Virtual Chassis.
show chassis fpc	Display status information about the installed Flexible PIC Concentrators (FPCs) and PICs.	(Default) Display status information for all FPCs on all members of the Virtual Chassis configuration.	Display status information for all FPCs on the local member of the Virtual Chassis.	Display status information for all FPCs on the specified member of the Virtual Chassis.
show chassis hardware	Display a list of all Flexible PIC Concentrators (FPCs) and PICs installed in the router or switch chassis, including the hardware version level and serial number.	(Default) Display hardware-specific information for all the members of the Virtual Chassis configuration.	Display hardware-specific information for the local member of the Virtual Chassis.	Display hardware-specific information for the specified member of the Virtual Chassis.

Table 13: Commands Available for Command Forwarding in an MX Series Virtual Chassis *(continued)*

Command	Purpose	all-members	local	member *member-id*
show chassis location	Display the physical location of the chassis.	(Default) Display the physical location of the chassis for all the member routers in the Virtual Chassis configuration.	Display the physical location of the chassis for the local member of the Virtual Chassis.	Display the physical location of the chassis for the specified member of the Virtual Chassis.
show chassis mac-addresses	Display the media access control (MAC) addresses for the router, switch chassis, or switch.	(Default) Display the MAC addresses for all the member routers of the Virtual Chassis configuration.	Display the MAC addresses for the local member of the Virtual Chassis.	Display the MAC addresses for the specified member of the Virtual Chassis.
show chassis pic	Display status information about the PIC installed in the specified Flexible PIC Concentrator (FPC) and PIC slot.	(Default) Display PIC information for all member routers in the Virtual Chassis configuration.	Display PIC information for the local member of the Virtual Chassis.	Display PIC information for the specified member of the Virtual Chassis.
show chassis power	Display power limits and usage information for the AC or DC power sources.	(Default) Display power usage information for all members of the Virtual Chassis configuration.	Display power usage information for the local member of the Virtual Chassis.	Display power usage information for the specified member of the Virtual Chassis.
show chassis routing-engine	Display the status of the Routing Engine.	(Default) Display Routing Engine information for all members of the Virtual Chassis configuration.	Display Routing Engine information for the local member of the Virtual Chassis.	Display Routing Engine information for the specified member of the Virtual Chassis.
show chassis sfb	(MX2010 and MX2020 routers only) Display chassis information about the Switch Fabric Boards (SFBs).	(Default) Display chassis information about the SFBs in all members of the Virtual Chassis configuration.	Display chassis information about the SFBs in the local member of the Virtual Chassis.	Display chassis information about the SFBs in the specified member of the Virtual Chassis.
show chassis spmb	(MX2010 and MX2020 routers only) Display status information for the Switch Processor Mezzanine Boards (SPMBs).	(Default) Display status information for the SPMBs in all members of the Virtual Chassis configuration.	Display status information for the SPMBs in the local member of the Virtual Chassis.	Display status information for the SPMBs in the specified member of the Virtual Chassis.
show chassis temperature-thresholds	Display chassis temperature threshold settings, in degrees Celsius.	(Default) Display the chassis temperature threshold settings of all member routers in the Virtual Chassis configuration.	Display the chassis temperature threshold settings of the local member of the Virtual Chassis.	Display the chassis temperature threshold settings of the specified member of the Virtual Chassis.

Table 13: Commands Available for Command Forwarding in an MX Series Virtual Chassis *(continued)*

Command	Purpose	all-members	local	member *member-id*
show chassis zones	(MX2010 and MX2020 routers only) Display the status of the cooling system zones of the chassis.	(Default) Display the status of the cooling system zones in all members of the Virtual Chassis configuration.	Display the status of the cooling system zones in the local member of the Virtual Chassis.	Display the status of the cooling system zones in the specified member of the Virtual Chassis.
show pfe fpc	Display Packet Forwarding Engine statistics for the specified Flexible PIC Concentrator (FPC).	(Default) Display Packet Forwarding Engine statistics for the specified FPC in all members of the Virtual Chassis configuration.	Display Packet Forwarding Engine statistics for the specified FPC in the local member of the Virtual Chassis.	Display Packet Forwarding Engine statistics for the specified FPC in the specified member of the Virtual Chassis.
show pfe terse	Display Packet Forwarding Engine status information.	(Default) Display Packet Forwarding Engine status information for all members in the Virtual Chassis configuration.	Display Packet Forwarding Engine status information for the local member of the Virtual Chassis.	Display Packet Forwarding Engine status information for the specified member of the Virtual Chassis.
show system audit	Display the state and checksum values for file systems.	(Default) Display file system MD5 hash and permissions information on all members of the Virtual Chassis configuration.	Display file system MD5 hash and permissions information on the local member of the Virtual Chassis.	Display file system MD5 hash and permissions information on the specified member of the Virtual Chassis.
show system boot-messages	Display initial messages generated by the system kernel upon startup.	(Default) Display boot time messages on all members of the Virtual Chassis configuration.	Display boot time messages on the local member of the Virtual Chassis.	Display boot time messages on the specified member of the Virtual Chassis.
show system buffers	Display information about the buffer pool that the Routing Engine uses for local traffic.	(Default) Show buffer statistics for all members of the Virtual Chassis configuration.	Show buffer statistics for the local member of the Virtual Chassis.	Show buffer statistics for the specified member of the Virtual Chassis.
show system connections	Display information about the active IP sockets on the Routing Engine.	(Default) Display system connection activity for all members of the Virtual Chassis configuration.	Display system connection activity for the local member of the Virtual Chassis.	Display system connection activity for the specified member of the Virtual Chassis.
show system directory-usage	Display directory usage information.	Display directory information for all members of the Virtual Chassis configuration.	(Default) Display directory information for the local member of the Virtual Chassis.	Display directory information for the specified member of the Virtual Chassis.

Table 13: Commands Available for Command Forwarding in an MX Series Virtual Chassis *(continued)*

Command	Purpose	all-members	local	member *member-id*
show system processes	Display information about software processes that are running on the router and that have controlling terminals.	(Default) Display standard system process information for all members of the Virtual Chassis configuration.	Display standard system process information for the local member of the Virtual Chassis.	Display standard system process information for the specified member of the Virtual Chassis.
show system queues	Display queue statistics.	(Default) Display system queue statistics for all members of the Virtual Chassis configuration.	Display system queue statistics for the local member of the Virtual Chassis.	Display system queue statistics for the specified member of the Virtual Chassis.
show system reboot	Display pending system reboots or halts.	(Default) Display halt or reboot request information for all members of the Virtual Chassis configuration.	Display halt or reboot request information for the local member of the Virtual Chassis.	Display halt or reboot request information for the specified member of the Virtual Chassis.
show system statistics	Display system-wide protocol-related statistics.	(Default) Display system statistics for a protocol for all members of the Virtual Chassis configuration.	Display system statistics for a protocol for the local member of the Virtual Chassis.	Display system statistics for a protocol for the specified member of the Virtual Chassis.
show system storage	Display statistics about the amount of free disk space in the router's file systems.	(Default) Display system storage statistics for all members of the Virtual Chassis configuration.	Display system storage statistics for the local member of the Virtual Chassis.	Display system storage statistics for the specified member of the Virtual Chassis.
show system switchover	Display whether graceful Routing Engine switchover is configured, the state of the kernel replication (ready or synchronizing), any replication errors, and whether the primary and standby Routing Engines are using compatible versions of the kernel database.	(Default) Display graceful Routing Engine switchover information for all Routing Engines on all members of the Virtual Chassis configuration.	Display graceful Routing Engines switchover information for all Routing Engines on the local member of the Virtual Chassis.	Display graceful Routing Engine switchover information for all Routing Engines on the specified member of the Virtual Chassis.
show system uptime	Display the current time and information about how long the router or switch, router or switch software, and routing protocols have been running.	(Default) Show time since the system rebooted and processes started on all members of the Virtual Chassis configuration.	Show time since the system rebooted and processes started on the local member of the Virtual Chassis.	Show time since the system rebooted and processes started on the specified member of the Virtual Chassis.

Table 13: Commands Available for Command Forwarding in an MX Series Virtual Chassis *(continued)*

Command	Purpose	all-members	local	member *member-id*
show system users	List information about the users who are currently logged in to the router.	(Default) Display users currently logged in to all members of the Virtual Chassis configuration.	Display users currently logged in to the local member of the Virtual Chassis.	Display users currently logged in to the specified member of the Virtual Chassis.
show system virtual-memory	Display the usage of Junos OS kernel memory listed first by size of allocation and then by type of usage.	(Default) Display kernel dynamic memory usage information for all members of the Virtual Chassis configuration.	Display kernel dynamic memory usage information for the local member of the Virtual Chassis.	Display kernel dynamic memory usage information for the specified member of the Virtual Chassis.
show version	Display the hostname and version information about the software running on the router.	(Default) Display standard information about the hostname and version of the software running on all members of the Virtual Chassis configuration.	Display standard information about the hostname and version of the software running on the local member of the Virtual Chassis.	Display standard information about the hostname and version of the software running on the specified member of the Virtual Chassis.
show version invoke-on	Display the hostname and version information about the software running on a router with two Routing Engines.	(Default) Display the hostname and version information about the software running on all master and backup Routing Engines on all members of the Virtual Chassis configuration.	Display the hostname and version information about the software running on all master and backup Routing Engines on the local member of the Virtual Chassis.	Display the hostname and version information about the software running on all master and backup Routing Engines on the specified member of the Virtual Chassis.

Related Documentation

- Virtual Chassis Components Overview on page 24
- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- CLI Explorer

CHAPTER 3

Configuring a Virtual Chassis

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- Preparing for a Virtual Chassis Configuration on page 60
- Installing Junos OS Licenses on Virtual Chassis Member Routers on page 62
- Creating and Applying Configuration Groups for a Virtual Chassis on page 64
- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Configuring Enhanced IP Network Services for a Virtual Chassis on page 69
- Enabling Graceful Routing Engine Switchover and Nonstop Active Routing for a Virtual Chassis on page 70
- Configuring Member IDs for a Virtual Chassis on page 72
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- Configuring an MX2020 Member Router in an Existing MX Series Virtual Chassis on page 92
- Switching the Global Master and Backup Roles in a Virtual Chassis Configuration on page 95
- Deleting Member IDs in a Virtual Chassis Configuration on page 97
- Disabling Split Detection in a Virtual Chassis Configuration on page 98
- Example: Replacing a Routing Engine in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 99
- Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 110
- Example: Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 111
- Upgrading an MX Virtual Chassis SCB or SCBE to SCBE2 on page 123

Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis

To provide a stateful interchassis redundancy solution for MX Series routers, you can configure a Virtual Chassis. A *Virtual Chassis* interconnects two MX Series routers into a logical system that you can manage as a single network element.

NOTE: For recommended settings on MX Virtual Chassis devices, please consult Maximizing Scaling and Performance for MX Series Virtual Chassis on the Juniper Networks Knowledge Base.

To configure a Virtual Chassis for MX Series routers:

1. Prepare your site for the Virtual Chassis configuration.

   See "Preparing for a Virtual Chassis Configuration" on page 60.

2. Install Junos OS licenses on the routers to be configured as members of the Virtual Chassis.

   See "Installing Junos OS Licenses on Virtual Chassis Member Routers" on page 62.

3. Define configuration groups for the Virtual Chassis.

   See "Creating and Applying Configuration Groups for a Virtual Chassis" on page 64.

4. Create the preprovisioned member configuration on the master router in the Virtual Chassis.

   See "Configuring Preprovisioned Member Information for a Virtual Chassis" on page 66.

5. Configure enhanced IP network services on both member routers.

   See "Configuring Enhanced IP Network Services for a Virtual Chassis" on page 69.

6. Enable graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) on both member routers.

   See "Enabling Graceful Routing Engine Switchover and Nonstop Active Routing for a Virtual Chassis" on page 70.

7. Set the preprovisioned member IDs and reboot the routers in Virtual Chassis mode.

   See "Configuring Member IDs for a Virtual Chassis" on page 72.

8. Create the Virtual Chassis ports to interconnect the member routers, and commit the Virtual Chassis configuration on the master router.

See "Configuring Virtual Chassis Ports to Interconnect Member Routers or Switches" on page 136.

9. (Optional) Verify the configuration and operation of the Virtual Chassis.

   See the following topics:

   - Verifying the Status of Virtual Chassis Member Routers or Switches on page 183
   - Verifying the Operation of Virtual Chassis Ports on page 183
   - Verifying Neighbor Reachability for Member Routers or Switches in a Virtual Chassis on page 184
   - Verifying Neighbor Reachability for Hardware Devices in a Virtual Chassis on page 184
   - Viewing Information in the Virtual Chassis Control Protocol Adjacency Database on page 186
   - Viewing Information in the Virtual Chassis Control Protocol Link-State Database on page 187
   - Viewing Information About Virtual Chassis Port Interfaces in the Virtual Chassis Control Protocol Database on page 188
   - Viewing Virtual Chassis Control Protocol Routing Tables on page 188
   - Viewing Virtual Chassis Control Protocol Statistics for Member Devices and Virtual Chassis Ports on page 189

Related Documentation

- Interchassis Redundancy and Virtual Chassis Overview on page 19
- Virtual Chassis Components Overview on page 24
- Guidelines for Configuring Virtual Chassis Ports on page 134
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Preparing for a Virtual Chassis Configuration

Before you configure and use an MX Series Virtual Chassis, we recommend that you prepare the hardware and software in your network for the configuration.

To prepare for configuring an MX Series Virtual Chassis:

1. Make a list of the serial numbers of each router that you want to configure as part of the Virtual Chassis.

   The chassis serial number is located on a label affixed to the side of the of the MX Series chassis. Alternatively, you can obtain the chassis serial number by issuing the **show chassis hardware** command, which is especially useful if you are accessing the router from a remote location. For example:

   ```
   user@gladius> show chassis hardware
   Hardware inventory:
   Item             Version  Part number  Serial number     Description
   Chassis                                JN10C7135AFC      MX240
   .
   .
   .
   ```

2. Note the desired function of each router in the Virtual Chassis.

   In a two-router Virtual Chassis configuration, you must designate each router with the **routing-engine** role, which enables either router to function as the master or backup of the Virtual Chassis.

   NOTE: **When configuring multiple Routing Engines in a Virtual Chassis, all must have the same amount of physical memory allocated.**

   - The *master router* maintains the global configuration and state information for all members of the Virtual Chassis, and runs the chassis management processes.
   - The *backup router* synchronizes with the master router and relays chassis control information (such as line-card presence and alarms) to the master router. If the master router is unavailable, the backup router takes mastership of the Virtual Chassis to preserve routing information and maintain network connectivity without disruption.

3. Note the member ID (0 or 1) to be assigned to each router in the Virtual Chassis.

4. Ensure that both MX Series routers in the Virtual Chassis have dual Routing Engines installed, and that all four Routing Engines in the Virtual Chassis are the same model.

   For example, you cannot configure a Virtual Chassis if one member router has RE-S-2000 Routing Engines installed and the other member router has RE-S-1800 Routing Engines installed.

   For the list of supported Routing Engines on MX series routers, see Supported Routing Engines by Router.

5. Ensure that the necessary Modular Port Concentrator/Modular Interface Card (MPC/MIC) interfaces on which to configure the Virtual Chassis ports are installed and operational in each router to be configured as a member of the Virtual Chassis.

   NOTE: An MX Series Virtual Chassis does not support a combination of 1-Gigabit Ethernet (ge media type) Virtual Chassis ports and 10-Gigabit Ethernet (xe media type) Virtual Chassis ports within the same Virtual Chassis. You must configure either all 10-Gigabit Ethernet Virtual Chassis ports or all 1-Gigabit Ethernet Virtual Chassis ports in the same Virtual Chassis. We recommend that you configure Virtual Chassis ports on 10-Gigabit Ethernet interfaces. This restriction has no effect on access ports or uplink ports in an MX Series Virtual Chassis configuration.

6. If MX Series Enhanced Queuing IP Services DPCs (DPCE-R-Q model numbers) or MX Series Enhanced Queuing Ethernet Services DPCs (DPCE-X-Q model numbers) are installed in a router to be configured as a member of the Virtual Chassis, make sure these DPCs are offline before you configure the Virtual Chassis. Otherwise, the MX Series Virtual Chassis configuration will not function.

   NOTE: MX Series Enhanced Queuing IP Services DPCs (DPCE-R-Q model numbers) and MX Series Enhanced Queuing Ethernet Services DPCs (DPCE-X-Q model numbers) do not interoperate with features of the MX Series Virtual Chassis.

7. Determine the desired location of the dedicated Virtual Chassis ports on both member routers, and use the Virtual Chassis ports to physically interconnect the member routers in a point-to-point topology.

8. Ensure that both MX Series routers to be configured as a member of the Virtual Chassis are running the same Junos OS release, and have basic network connectivity.

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- Guidelines for Configuring Virtual Chassis Ports on page 134
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Installing Junos OS Licenses on Virtual Chassis Member Routers

To enable some Junos OS features or router scaling levels, you might have to purchase, install, and manage separate software license packs. The presence on the router of the appropriate software license keys (passwords) determines whether you can configure and use certain features or configure a feature to a predetermined scale.

Before you configure an MX Series Virtual Chassis, install the following Junos OS software licenses on each MX Series router to be configured as a member of the Virtual Chassis:

- MX Virtual Chassis Redundancy Feature Pack—You must purchase and install a unique MX Virtual Chassis Redundancy Feature Pack for each member router in the Virtual Chassis. If you issue the **request virtual-chassis member-id set**, **request virtual-chassis member-id delete**, **request virtual-chassis vc-port set**, or **request virtual-chassis vc-port delete** command to set or delete member IDs or Virtual Chassis ports without first installing an MX Virtual Chassis Redundancy Feature Pack on both member routers, the software displays a warning message that you are operating without a valid Virtual Chassis software license.
- Junos OS feature licenses—Purchase and install the appropriate Junos OS feature licenses to enable use of a particular software feature or scaling level in your network. You must install the required feature licenses on each member router in the Virtual Chassis.

Before you begin:

- Prepare your site for the Virtual Chassis configuration.

  See "Preparing for a Virtual Chassis Configuration" on page 60.
- Familiarize yourself with the procedures for installing and managing Junos OS licenses.

  See *Installation and Upgrade Guide*.

To install Junos OS licenses on each member router in the Virtual Chassis:

1. Install the required licenses on the MX Series router to be designated as the protocol master for the Virtual Chassis.
   a. Install the MX Virtual Chassis Redundancy Feature Pack.
   b. Install the Junos OS feature licenses required for your software feature or scaling level.
2. Install the required licenses on the MX Series router to be designated as the protocol backup for the Virtual Chassis.
   a. Install the MX Virtual Chassis Redundancy Feature Pack.
   b. Install the Junos OS feature licenses required for your software feature or scaling level.
3. (Optional) Verify the license installation on each member router.

For example:

```
user@host> show system license
License usage:
                                Licenses     Licenses    Licenses    Expiry
  Feature name                       used    installed      needed
  subscriber-accounting                 0            1           0    permanent
  subscriber-authentication             0            1           0    permanent
  subscriber-address-assignment         0            1           0    permanent
  subscriber-vlan                       0            1           0    permanent
  subscriber-ip                         0            1           0    permanent
  scale-subscriber                      0       256000           0    permanent
  scale-l2tp                            0         1000           0    permanent
  scale-mobile-ip                       0         1000           0    permanent
  virtual-chassis                       0            1           0    permanent
```

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- *Software Features That Require Licenses on MX Series Routers Only*
- *Installation and Upgrade Guide*

Creating and Applying Configuration Groups for a Virtual Chassis

For a Virtual Chassis configuration consisting of two MX Series routers or two EX9200 switches, each of which supports dual Routing Engines, you must create and apply on the master device of the Virtual Chassis the following configuration groups, instead of using the standard **re0** and **re1** configuration groups:

- **member0-re0**
- **member0-re1**
- **member1-re0**
- **member1-re1**

NOTE: The member*n*-re*n* naming format for configuration groups is reserved for exclusive use by member routers or switches in EX9200 or MX Series Virtual Chassis configurations.

Using configuration group names of the form member*n*-re*n* in an existing non-Virtual Chassis configuration or configuration script could interfere with Virtual Chassis operation. This misconfiguration could cause the router or switch to assign no IP address or an incorrect IP address to the fxp0 management Ethernet interface, and could result in a display of the Amnesiac prompt during login.

To create and apply configuration group information from the router or switch to be configured as the master of the Virtual Chassis:

1. In the console window on the master router or switch (**member 0** in this procedure), create and apply the **member0-re0** configuration group.

   ```
   [edit]
   user@host# copy groups re0 to member0-re0
   user@host# set apply-groups member0-re0
   ```

2. Delete the standard **re0** configuration group from the global configuration on **member 0**.

   ```
   [edit]
   user@host# delete apply-groups re0
   user@host# delete groups re0
   ```

3. Create and apply the **member0-re1** configuration group.

   ```
   [edit]
   user@host# copy groups re1 to member0-re1
   user@host# set apply-groups member0-re1
   ```

4. Delete the standard **re1** configuration group from the global configuration on **member 0**.

```
[edit]
user@host# delete apply-groups re1
user@host# delete groups re1
```

5. Create and apply the **member1-re0** configuration information.

```
[edit]
user@host# set groups member1-re0 system host-name host-name
user@host# set groups member1-re0 system backup-router address
user@host# set groups member1-re0 system backup-router destination
  destination-address
user@host# set groups member1-re0 system backup-router destination
  destination-address
...
user@host# set groups member1-re0 interfaces fxp0 unit unit-number family inet
  address address
user@host# set apply-groups member1-re0
```

The commands in Steps 5 and 6 set the IP address for the **fxp0** management interface and add an IP route for it in the event that routing becomes inactive.

6. Create and apply the **member1-re1** configuration information.

```
[edit]
user@host# set groups member1-re1 system host-name host-name
user@host# set groups member1-re1 system backup-router address
user@host# set groups member1-re1 system backup-router destination
  destination-address
user@host# set groups member1-re1 system backup-router destination
  destination-address
...
user@host# set groups member1-re1 interfaces fxp0 unit unit-number family inet
  address address
user@host# set apply-groups member1-re1
```

7. Commit the configuration.

BEST PRACTICE: We recommend that you use the **commit synchronize** command to save any configuration changes to the Virtual Chassis.

For an EX9200 or MX Series Virtual Chassis, the **force** option is the default and only behavior when you issue the **commit synchronize** command. Issuing the **commit synchronize** command for a Virtual Chassis configuration has the same effect as issuing the **commit synchronize force** command.

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58

- *Configuring an EX9200 Virtual Chassis*
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- For more information about creating and managing configuration groups, see the *Junos OS CLI User Guide*

Configuring Preprovisioned Member Information for a Virtual Chassis

To configure a Virtual Chassis for MX Series routers, you must create a preprovisioned configuration on the master router by including the **virtual-chassis** stanza at the **[edit virtual-chassis]** hierarchy level. The preprovisioned configuration specifies the chassis serial number, member ID, and role for both member routers in the Virtual Chassis.

When a new member router joins the Virtual Chassis, the software compares its serial number against the values specified in the preprovisioned configuration. If the serial number of a joining router does not match any of the configured serial numbers, the software prevents that router from becoming a member of the Virtual Chassis.

To configure the preprovisioned member information for an MX Series Virtual Chassis:

1. Specify that you want to create a preprovisioned Virtual Chassis configuration.

   ```
   [edit virtual-chassis]
   user@host# set preprovisioned
   ```

2. Configure the member ID (**0** or **1**), role (**routing-engine**), and chassis serial number for each member router in the Virtual Chassis.

   ```
   [edit virtual-chassis]
   user@host# set member member-number role routing-engine serial-number
     serial-number
   user@host# set member member-number role routing-engine serial-number
     serial-number
   ```

   NOTE: In a two-member MX Series Virtual Chassis configuration, you must assign the routing-engine role to each router. The routing-engine role enables the router to function either as the master router or backup router of the Virtual Chassis.

3. Disable detection of a split in the Virtual Chassis configuration. (By default, split detection in an MX Series Virtual Chassis is enabled.)

   ```
   [edit virtual-chassis]
   user@host# set no-split-detection
   ```

   BEST PRACTICE: We recommend that you disable split detection for a two-member MX Series Virtual Chassis configuration if you think the

backup router is more likely to fail than the Virtual Chassis port links to the backup router. Configuring redundant Virtual Chassis ports on different line cards in each member router reduces the likelihood that all Virtual Chassis port links to the backup router will fail.

4. (Optional) Enable locality bias in the Virtual Chassis configuration.

```
[edit virtual-chassis]
user@host# set locality-bias
```

BEST PRACTICE: Starting in Junos OS Release 14.1, you can enable locality bias in the Virtual Chassis configuration. Locality bias can cause traffic loss and oversubscription on egress interfaces if you configure it in a network that is not designed to handle locality biasing. Make sure you understand the utilization requirements, such as total and available bandwidth, for the local links in your network before changing the locality bias configuration.

5. (Optional) Enable tracing of Virtual Chassis operations.

   For example:

```
[edit virtual-chassis]
user@gladius# set traceoptions file filename
user@gladius# set traceoptions file size maximum-file-size
user@gladius# set traceoptions flag flag
```

6. Commit the configuration.

BEST PRACTICE: We recommend that you use the **commit synchronize** command to save any configuration changes to the Virtual Chassis.

For an MX Series Virtual Chassis, the **force** option is the default and only behavior when you issue the **commit synchronize** command. Issuing the **commit synchronize** command for an MX Series Virtual Chassis configuration has the same effect as issuing the **commit synchronize force** command.

The following example shows an MX Series Virtual Chassis preprovisioned configuration for two member routers.

```
[edit virtual-chassis]
user@gladius# show
preprovisioned;
no-split-detection;
locality-bias;
```

```
traceoptions {
    file vccp size 10m;
    flag all;
}
member 0 {
    role routing-engine;
    serial-number JN115FDADAFB;
}
member 1 {
    role routing-engine;
    serial-number JN10C78D1AFC;
}
```

Release History Table

Release	Description
14.1	Starting in Junos OS Release 14.1, you can enable locality bias in the Virtual Chassis configuration.

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Configuring Enhanced IP Network Services for a Virtual Chassis

For an existing MX Series Virtual Chassis to function properly, you must configure enhanced IP network services on all member routers in the Virtual Chassis from the Virtual Chassis master router.

Enhanced IP network services defines how the chassis recognizes and uses certain modules. When you set each member router's network services to **enhanced-ip**, only MPC/MIC modules and MS-DPC modules are powered on in the chassis. Non-service DPCs do not work with enhanced IP network services.

This procedure describes how to configure enhanced IP network services for an existing MX Series Virtual Chassis. For information about configuring enhanced IP network services when you first set up the Virtual Chassis, see *Configuring Enhanced IP Network Services* in "Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis" on page 75.

BEST PRACTICE: We recommend that you use the **commit synchronize** command to save any configuration changes to the Virtual Chassis.

For an MX Series Virtual Chassis, the **force** option is the default and only behavior when you issue the **commit synchronize** command. Issuing the **commit synchronize** command for an MX Series Virtual Chassis configuration has the same effect as issuing the **commit synchronize force** command.

To configure enhanced IP network services for an existing Virtual Chassis:

1. Log in to the console for the master Routing Engine in the Virtual Chassis master router (member0-re0 in this procedure).

2. Access the chassis hierarchy.

   ```
   {master:member0-re0}[edit]
   user@hostA# edit chassis
   ```

3. Configure enhanced IP network services on member 0.

   ```
   {master:member0-re0}[edit chassis]
   user@hostA# set network-services enhanced-ip
   ```

4. Commit the configuration.

5. When prompted to do so, reboot all Routing Engines in the Virtual Chassis.

   ```
   {master:member0-re0}
   user@hostA> request system reboot
   ```

The **request system reboot** command reboots both Routing Engines in each member router forming the Virtual Chassis.

6. (Optional) Verify that enhanced IP network services has been properly configured for the Virtual Chassis.

   a. Verify that enhanced IP network services is configured on the master Routing Engine in the Virtual Chassis master router (member0-re0).

      ```
      {master:member0-re0}
      user@hostA> show chassis network-services

      Network Services Mode: Enhanced-IP
      ```

   b. Verify that enhanced IP network services is configured on the master Routing Engine in the Virtual Chassis backup router (member1-re0).

      ```
      {backup:member1-re0}
      user@hostB> show chassis network-services

      Network Services Mode: Enhanced-IP
      ```

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Enabling Graceful Routing Engine Switchover and Nonstop Active Routing for a Virtual Chassis

Before you configure member IDs and Virtual Chassis ports, you must enable graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) on both member routers in the Virtual Chassis.

To enable graceful Routing Engine switchover and nonstop active routing:

1. Enable graceful Routing Engine switchover and nonstop active routing on member 0 (**gladius**):

   a. Log in to the console on member 0.

   b. Enable graceful switchover.

      ```
      [edit chassis redundancy]
      user@gladius# set graceful-switchover
      ```

   c. Enable nonstop active routing.

      ```
      [edit routing-options]
      user@gladius# set nonstop-routing
      ```

   d. Configure the **commit** command to automatically result in a **commit synchronize** action between the dual Routing Engines in member 0.

```
[edit system]
user@gladius# set commit synchronize
```

e. Commit the configuration.

2. Enable graceful Routing Engine switchover and nonstop active routing on member 1 (**trefoil**):

   a. Log in to the console on member 1.

   b. Enable graceful switchover.

   ```
   [edit chassis redundancy]
   user@trefoil# set graceful-switchover
   ```

   c. Enable nonstop active routing.

   ```
   [edit routing-options]
   user@trefoil# set nonstop-routing
   ```

   d. Configure the **commit** command to automatically result in a **commit synchronize** action between the dual Routing Engines in member 1.

   ```
   [edit system]
   user@trefoil# set commit synchronize
   ```

   e. Commit the configuration.

NOTE: When you configure nonstop active routing, you must include the **commit synchronize** statement at the **[edit system]** hierarchy level. Otherwise, the commit operation fails.

For an MX Series Virtual Chassis, the **force** option is the default and only behavior when you use the **commit synchronize** statement. Including the **commit synchronize** statement for an MX Series Virtual Chassis configuration has the same effect as including the **commit synchronize force** statement.

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- *Configuring Graceful Routing Engine Switchover*
- *Configuring Nonstop Active Routing*

Configuring Member IDs for a Virtual Chassis

After you commit the preprovisioned configuration on the master router, you must assign the preprovisioned member IDs to both MX Series routers in the Virtual Chassis by using the **request virtual-chassis member-id set** command. In an MX Series Virtual Chassis, you can optionally include the **slots-per-chassis *slot-count*** option to identify the number of chassis slots in the member router. Assigning the member ID and, optionally, ***slot-count*** causes the router to reboot in preparation for forming the Virtual Chassis.

NOTE: If you issue the **request virtual-chassis member-id set** command without first installing an MX Virtual Chassis Redundancy Feature Pack license on both member routers, the software displays a warning message that you are operating without a valid Virtual Chassis software license.

For information about the supported member router combinations in an MX Series Virtual Chassis, see "Interchassis Redundancy and Virtual Chassis Overview" on page 19. Platform support depends on the Junos OS release in your installation.

To configure the member ID and, optionally, slot count for each member router in an MX Series Virtual Chassis:

1. Set the member ID on the router configured as member 0 in one of the following ways:

   - For a Virtual Chassis consisting of a combination of MX240 routers, MX480 routers, and MX960 routers:

     user@hostA> request virtual-chassis member-id set **member 0**

     ```
     This command will enable virtual-chassis mode and reboot the system.
     Continue? [yes,no] yes
     ```

     With this Virtual Chassis configuration, the default ***slot-count*** (12) is the same for both member routers, so you do not need to configure it.

   - For a Virtual Chassis consisting of an MX2010 router and either an MX960 router or MX2010 router:

     user@hostA> request virtual-chassis member-id set **member 0**

     ```
     This command will enable virtual-chassis mode and reboot the system.
     Continue? [yes,no] yes
     ```

     With this Virtual Chassis configuration, the default ***slot-count*** (12) is the same for an MX960 router and an MX2010 router, so you do not need to configure it.

   - For a Virtual Chassis consisting of an MX2020 router and either an MX960 router or MX2010 router:

     user@hostA> request virtual-chassis member-id set **member 0 slots-per-chassis *slot-count***

     ```
     This command will enable virtual-chassis mode and reboot the system.
     Continue? [yes,no] yes
     ```

     To ensure that this Virtual Chassis configuration forms properly, you *must* set the ***slot-count*** value of the MX960 router or MX2010 router to 20 to match the ***slot-count*** of the MX2020 router. For example:

     user@hostA> request virtual-chassis member-id set **member 0 slots-per-chassis 20**

   - For a Virtual Chassis consisting of two MX2020 routers:

     user@hostA> request virtual-chassis member-id set **member 0**

     ```
     This command will enable virtual-chassis mode and reboot the system.
     Continue? [yes,no] yes
     ```

     With this Virtual Chassis configuration, the default ***slot-count*** (20) is the same for both MX2020 routers, so you do not need to configure it.

   The router reboots in preparation for forming the Virtual Chassis. After the reboot, all MPCs remain powered off until the Virtual Chassis port connection is configured.

2. Repeat Step 1 to set the member ID on the router configured as **member 1**.

   user@hostB> request virtual-chassis member-id set **member 1**

```
This command will enable virtual-chassis mode and reboot the system.
 Continue? [yes,no] yes
```

or

user@hostB> request virtual-chassis member-id set **member 1 slots-per-chassis** ***slot-count***

```
This command will enable virtual-chassis mode and reboot the system.
 Continue? [yes,no] yes
```

For example:

user@hostA> request virtual-chassis member-id set **member 0 slots-per-chassis 20**

The router reboots in preparation for forming the Virtual Chassis. After the reboot, all MPCs remain powered off until the Virtual Chassis port connection is configured.

3. (Optional) Verify the member ID configuration for **member 0**.

   For example:

```
{master:member0-re0}
user@hostA> show virtual-chassis status
Preprovisioned Virtual Chassis
Virtual Chassis ID: 4f2b.1aa0.de08
                                                Mastership           Neighbor
List
Member ID       Status   Serial No    Model    priority    Role      ID
Interface

0 (FPC   0- 11) Prsnt    JN10C7135AFC mx240          129  Master*
```

4. (Optional) Verify the member ID configuration for **member 1**.

   For example:

```
Amnesiac (ttyd0)

login: user
Password:
...
{master:member1-re0}
user> show virtual-chassis status
Virtual Chassis ID: ef98.2c6c.f7f7
                                                Mastership           Neighbor
List
Member ID       Status   Serial No    Model    priority    Role      ID
Interface

1 (FPC  12- 23) Prsnt    JN115D117AFB mx480          128  Master*
```

NOTE: At this point in the configuration procedure, all line cards are offline, and the routers are each designated with the Master role because they are not yet interconnected as a fully formed Virtual Chassis. In addition, member 1 remains in Amnesiac state (has no defined configuration) until the Virtual Chassis forms and the configuration is committed.

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- Interchassis Redundancy and Virtual Chassis Overview on page 19

Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis

To provide interchassis redundancy for MX Series 3D Universal Edge Routers, you can configure a Virtual Chassis. A *Virtual Chassis* configuration interconnects two MX Series routers into a logical system that you can manage as a single network element. The member routers in a Virtual Chassis are interconnected by means of Virtual Chassis ports that you configure on Modular Port Concentrator/Modular Interface Card (MPC/MIC) interfaces (network ports) on each MX Series router.

This example describes how to set up and configure a Virtual Chassis consisting of two MX Series routers:

- Requirements on page 75
- Overview and Topology on page 76
- Configuration on page 78
- Verification on page 89

Requirements

This example uses the following software and hardware components:

- Junos OS Release 11.2 and later releases
- One MX240 3D Universal Edge Router
- One MX480 3D Universal Edge Router

NOTE: This configuration example has been tested using the software release listed and is assumed to work on all later releases.

See Table 14 on page 77 for information about the hardware installed in each MX Series router.

BEST PRACTICE: We recommend that you use the **commit synchronize** command throughout this procedure to save any configuration changes to the Virtual Chassis.

For an MX Series Virtual Chassis, the **force** option is the default and only behavior when you issue the **commit synchronize** command. Issuing the **commit synchronize** command for an MX Series Virtual Chassis configuration has the same effect as issuing the **commit synchronize force** command.

Overview and Topology

To configure the Virtual Chassis shown in this example, you must create a preprovisioned configuration at the **[edit virtual-chassis]** hierarchy level on the router to be designated as the master of the Virtual Chassis. The preprovisioned configuration includes the serial number, member ID, and role for each member router (also known as member chassis) in the Virtual Chassis. When a new member router joins the Virtual Chassis, the software compares its serial number against the values specified in the preprovisioned configuration. If the serial number of a joining router does not match any of the configured serial numbers, the software prevents that router from becoming a member of the Virtual Chassis.

After you commit the preprovisioned configuration on the master router, you must assign the preprovisioned member IDs by issuing the **request virtual-chassis member-id set** administrative command on each router, which causes the router to reboot. When the reboot is complete, you create one or more Virtual Chassis ports by issuing the **request virtual-chassis vc-port set** administrative command on each router. The Virtual Chassis forms when the line cards in both member routers are back online.

This example configures a Virtual Chassis that interconnects two MX Series routers, and uses the basic topology shown in Figure 2 on page 76. For redundancy, two Virtual Chassis ports are configured on each member router.

Figure 2: Sample Topology for a Virtual Chassis with Two MX Series Routers

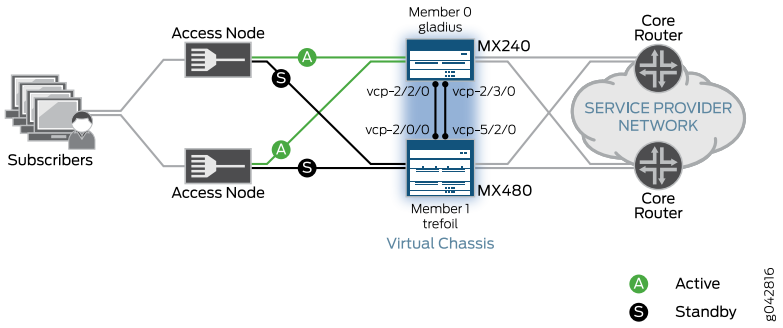


Table 14 on page 77 shows the hardware and software configuration settings for each MX Series router in the Virtual Chassis. You use some of these settings in the preprovisioned configuration and when you assign the member IDs and create the Virtual Chassis ports.

NOTE: MX Series Enhanced Queuing IP Services DPCs (DPCE-R-Q model numbers) and MX Series Enhanced Queuing Ethernet Services DPCs (DPCE-X-Q model numbers) do not interoperate with features of the MX Series Virtual Chassis. If any MX Series Enhanced Queuing DPCs are installed in a router to be configured as a member of a Virtual Chassis, you must ensure that these DPCs are offline before you configure the Virtual Chassis.

Table 14: Components of the Sample MX Series Virtual Chassis

Router Name	Hardware	Serial Number	Member ID	Role	Virtual Chassis Ports	Network Port Slot Numbering
gladius	MX240 router with: • 60-Gigabit Ethernet Enhanced Queuing MPC • 20-port Gigabit Ethernet MIC with SFP • 4-port 10-Gigabit Ethernet MIC with XFP • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member0-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member0-re1**)	JN10C7135AFC	0	routing-engine (master)	vcp-2/2/0 vcp-2/3/0	FPC 0 – 11

Table 14: Components of the Sample MX Series Virtual Chassis *(continued)*

Router Name	Hardware	Serial Number	Member ID	Role	Virtual Chassis Ports	Network Port Slot Numbering
trefoil	MX480 router with: • Two 30-Gigabit Ethernet Queuing MPCs • Two 20-port Gigabit Ethernet MICs with SFP • Two 2-port 10-Gigabit Ethernet MICs with XFP • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member1-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member1-re1**)	JN115D117AFB	1	routing-engine (backup)	vcp-2/0/0 vcp-5/2/0	FPC 12 – 23 (offset = 12)

Configuration

To configure a Virtual Chassis consisting of two MX Series routers, perform these tasks:

- Preparing for the Virtual Chassis Configuration on page 79
- Creating and Applying Configuration Groups for the Virtual Chassis on page 81
- Configuring Preprovisioned Member Information for the Virtual Chassis on page 83
- Configuring Enhanced IP Network Services on page 84
- Enabling Graceful Routing Engine Switchover and Nonstop Active Routing on page 85
- Configuring Member IDs and Rebooting the Routers to Enable Virtual Chassis Mode on page 86
- Configuring Virtual Chassis Ports to Interconnect Member Routers on page 88

Preparing for the Virtual Chassis Configuration

Step-by-Step Procedure

To prepare for configuring an MX Series Virtual Chassis:

1. Make a list of the serial numbers of both routers that you want to configure as part of the Virtual Chassis.

   The chassis serial number is located on a label affixed to the side of the of the MX Series chassis. Alternatively, you can obtain the chassis serial number by issuing the **show chassis hardware** command, which is especially useful if you are accessing the router from a remote location. For example:

   ```
   user@gladius> show chassis hardware
   Hardware inventory:
   Item             Version  Part number  Serial number     Description
   Chassis                                JN10C7135AFC      MX240
   .
   .
   .
   Fan Tray 0       REV 01   710-021113   JT0119            MX240 Fan Tray
   ```

2. Note the desired role (**routing-engine**) for each router in the Virtual Chassis.

   In a two-router Virtual Chassis configuration, you must designate each router with the **routing-engine** role, which enables either router to function as the master or backup of the Virtual Chassis.

   - The *master router* maintains the global configuration and state information for all members of the Virtual Chassis, and runs the chassis management processes.
   - The *backup router* synchronizes with the master router and relays chassis control information (such as line-card presence and alarms) to the master router. If the master router is unavailable, the backup router takes mastership of the Virtual Chassis to preserve routing information and maintain network connectivity without disruption.

3. Note the member ID (0 or 1) to be assigned to each router in the Virtual Chassis.

   In this example, the master router is assigned member ID 0, and the backup router is assigned member ID 1.

4. Ensure that both MX Series routers in the Virtual Chassis have dual Routing Engines installed, and that all four Routing Engines in the Virtual Chassis are the same model.

   For example, you cannot configure a Virtual Chassis if one member router has RE-S-2000 Routing Engines installed and the other member router has RE-S-1800 Routing Engines installed.

5. Ensure that the necessary Modular Port Concentrator/Modular Interface Card (MPC/MIC) interfaces on which to configure the Virtual Chassis ports are installed and operational in each router to be configured as a member of the Virtual Chassis.

NOTE: An MX Series Virtual Chassis does not support a combination of 1-Gigabit Ethernet (ge media type) Virtual Chassis ports and 10-Gigabit Ethernet (xe media type) Virtual Chassis ports within the same Virtual Chassis. You must configure either all 10-Gigabit Ethernet Virtual Chassis ports or all 1-Gigabit Ethernet Virtual Chassis ports in the same Virtual Chassis. We recommend that you configure Virtual Chassis ports on 10-Gigabit Ethernet interfaces. This restriction has no effect on access ports or uplink ports in an MX Series Virtual Chassis configuration.

6. If MX Series Enhanced Queuing IP Services DPCs (DPCE-R-Q model numbers) or MX Series Enhanced Queuing Ethernet Services DPCs (DPCE-X-Q model numbers) are installed in a router to be configured as a member of the Virtual Chassis, make sure these DPCs are offline before you configure the Virtual Chassis. Otherwise, the MX Series Virtual Chassis configuration will not function.

NOTE: MX Series Enhanced Queuing IP Services DPCs (DPCE-R-Q model numbers) and MX Series Enhanced Queuing Ethernet Services DPCs (DPCE-X-Q model numbers) do not interoperate with features of the MX Series Virtual Chassis.

7. Determine the desired location of the dedicated Virtual Chassis ports on both member routers, and use the Virtual Chassis ports to physically interconnect the member routers in a point-to-point topology.

8. Ensure that both MX Series routers to be configured as members of the Virtual Chassis are running the same Junos OS release, and have basic network connectivity.

9. Install the MX Virtual Chassis Redundancy Feature Pack license on each router to be configured as part of the Virtual Chassis.

10. Install the necessary Junos OS feature licenses on each router to be configured as part of the Virtual Chassis.

Creating and Applying Configuration Groups for the Virtual Chassis

Step-by-Step Procedure

For a Virtual Chassis configuration consisting of two MX Series routers, each of which supports dual Routing Engines, you must create and apply the following configuration groups on the router to be designated as the master of the Virtual Chassis instead of using the standard re0 and re1 configuration groups:

- **member0-re0**
- **member0-re1**
- **member1-re0**
- **member1-re1**

NOTE: The **member*n*-re*n*** naming format for configuration groups is reserved for exclusive use by member routers in MX Series Virtual Chassis configurations.

To create and apply configuration group information for the Virtual Chassis:

1. Log in to the console on member 0 (**gladius**).

2. In the console window on member 0, create and apply the **member0-re0** configuration group.

```
[edit]
user@gladius# copy groups re0 to member0-re0
user@gladius# set apply-groups member0-re0
```

3. Delete the standard **re0** configuration group from the global configuration on member 0.

```
[edit]
user@gladius# delete apply-groups re0
user@gladius# delete groups re0
```

4. Create and apply the **member0-re1** configuration group on member 0.

```
[edit]
user@gladius# copy groups re1 to member0-re1
user@gladius# set apply-groups member0-re1
```

5. Delete the standard **re1** configuration group from the global configuration on member 0.

```
[edit]
user@gladius# delete apply-groups re1
user@gladius# delete groups re1
```

6. Create and apply the **member1-re0** configuration information on member 0.

```
[edit]
user@gladius# set groups member1-re0 system host-name trefoil
user@gladius# set groups member1-re0 system backup-router 10.9.0.1
user@gladius# set groups member1-re0 system backup-router destination
  172.16.0.0/12
user@gladius# set groups member1-re0 system backup-router destination
  10.9.0.0/16
...
user@gladius# set groups member1-re0 interfaces fxp0 unit 0 family inet address
  10.9.3.97/21
user@gladius# set apply-groups member1-re0
```

   The examples in Steps 5 and 6 set the IP address for the **fxp0** management interface and add an IP route for it in the event that routing becomes inactive.

7. Create and apply the **member1-re1** configuration information on member 0.

```
[edit]
user@gladius# set groups member1-re1 system host-name trefoil
user@gladius# set groups member1-re1 system backup-router 10.9.0.1
user@gladius# set groups member1-re1 system backup-router destination
  172.16.0.0/12
user@gladius# set groups member1-re1 system backup-router destination 10.9.0.0/16
...
user@gladius# set groups member1-re1 interfaces fxp0 unit 0 family inet address
  10.9.3.98/21
user@gladius# set apply-groups member1-re1
```

8. Commit the configuration on member 0.

Results Display the results of the configuration.

```
[edit]
user@gladius# show groups ?
  Possible completions:
    <[Enter]>            Execute this command
    <group_name>         Group name
    global               Group name
    member0-re0          Group name
    member0-re1          Group name
    member1-re0          Group name
    member1-re1          Group name
    |                    Pipe through a command
```

```
[edit]
user@gladius# show apply-groups
  apply-groups [ global member0-re0 member0-re1 member1-re0 member1-re1 ];
```

Configuring Preprovisioned Member Information for the Virtual Chassis

Step-by-Step Procedure

To configure the preprovisioned member information on member 0 (**gladius**):

1. Log in to the console on member 0.

2. Specify that you want to create a preprovisioned Virtual Chassis configuration.

```
[edit virtual-chassis]
user@gladius# set preprovisioned
```

3. Configure the member ID (**0** or **1**), role (**routing-engine**), and chassis serial number for each member router in the Virtual Chassis.

```
[edit virtual-chassis]
user@gladius# set member 0 role routing-engine serial-number JN10C7135AFC
user@gladius# set member 1 role routing-engine serial-number JN115D117AFB
```

4. Disable detection of a split in the Virtual Chassis configuration. (By default, split detection in an MX Series Virtual Chassis is enabled.)

```
[edit virtual-chassis]
user@gladius# set no-split-detection
```

BEST PRACTICE: We recommend that you disable split detection for a two-member MX Series Virtual Chassis configuration if you think the backup router is more likely to fail than the Virtual Chassis port links to the backup router. Configuring redundant Virtual Chassis ports on different line cards in each member router reduces the likelihood that all Virtual Chassis port links to the backup router will fail.

5. (Optional) Enable tracing of Virtual Chassis operations.

```
[edit virtual-chassis]
user@gladius# set traceoptions file vccp
user@gladius# set traceoptions file size 100m
user@gladius# set traceoptions flag all
```

6. Commit the configuration.

Results

Display the results of the configuration.

```
[edit virtual-chassis]
user@gladius# show
preprovisioned;
no-split-detection;
traceoptions {
```

```
    file vccp size 100m;
    flag all;
  }
  member 0 {
    role routing-engine;
    serial-number JN10C7135AFC;
  }
  member 1 {
    role routing-engine;
    serial-number JN115D117AFB;
  }
```

Configuring Enhanced IP Network Services

Step-by-Step Procedure

For an MX Series Virtual Chassis to function properly, you must configure enhanced IP network services on both member routers (member 0 and member 1). Enhanced IP network services defines how the chassis recognizes and uses certain modules. When you set each member router's network services to **enhanced-ip**, only MPC/MIC modules and MS-DPC modules are powered on in the chassis. Non-service DPCs do not work with enhanced IP network services.

This procedure describes how to configure enhanced IP network services when you first set up the Virtual Chassis. For information about configuring enhanced IP network services for an existing MX Series Virtual Chassis, see "Configuring Enhanced IP Network Services for a Virtual Chassis" on page 69.

To configure enhanced IP network services for a Virtual Chassis:

1. Configure enhanced IP network services on member 0 (**gladius**).

   a. Log in to the console on member 0.

   b. Access the chassis hierarchy.

      ```
      [edit]
      user@gladius# edit chassis
      ```

   c. Configure enhanced IP network services for member 0.

      ```
      [edit chassis]
      user@gladius# set network-services enhanced-ip
      ```

   d. Commit the configuration on member 0.

NOTE: Immediately after you commit the configuration, the software prompts you to reboot the router. You can proceed without rebooting the router at this point because a reboot occurs when you configure the member IDs to enable Virtual Chassis mode, later in this procedure.

2. Configure enhanced IP network services on member 1 (**trefoil**).

a. Log in to the console on member 1.

b. Access the chassis hierarchy.

```
[edit]
user@trefoil# edit chassis
```

c. Configure enhanced IP network services for member 1.

```
[edit chassis]
user@trefoil# set network-services enhanced-ip
```

d. Commit the configuration on member 1.

NOTE: Immediately after you commit the configuration, the software prompts you to reboot the router. You can proceed without rebooting the router at this point because a reboot occurs when you configure the member IDs to enable Virtual Chassis mode, later in this procedure.

Enabling Graceful Routing Engine Switchover and Nonstop Active Routing

Step-by-Step Procedure

Before you configure member IDs and Virtual Chassis ports, you must enable graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) on both member routers in the Virtual Chassis.

To enable graceful Routing Engine switchover and nonstop active routing:

1. Enable graceful Routing Engine switchover and nonstop active routing on member 0 (**gladius**):

   a. Log in to the console on member 0.

   b. Enable graceful switchover.

   ```
   [edit chassis redundancy]
   user@gladius# set graceful-switchover
   ```

   c. Enable nonstop active routing.

   ```
   [edit routing-options]
   user@gladius# set nonstop-routing
   ```

   d. Configure the **commit** command to automatically result in a **commit synchronize** action between the dual Routing Engines in member 0.

   ```
   [edit system]
   user@gladius# set commit synchronize
   ```

   e. Commit the configuration.

2. Enable graceful Routing Engine switchover and nonstop active routing on member 1 (**trefoil**):

a. Log in to the console on member 1.

b. Enable graceful switchover.

```
[edit chassis redundancy]
user@trefoil# set graceful-switchover
```

c. Enable nonstop active routing.

```
[edit routing-options]
user@trefoil# set nonstop-routing
```

d. Configure the **commit** command to automatically result in a **commit synchronize** action between the dual Routing Engines in member 1.

```
[edit system]
user@trefoil# set commit synchronize
```

e. Commit the configuration.

NOTE: When you configure nonstop active routing, you must include the **commit synchronize** statement at the **[edit system]** hierarchy level. Otherwise, the commit operation fails.

For an MX Series Virtual Chassis, the **force** option is the default and only behavior when you use the **commit synchronize** statement. Including the **commit synchronize** statement for an MX Series Virtual Chassis configuration has the same effect as including the **commit synchronize force** statement.

Configuring Member IDs and Rebooting the Routers to Enable Virtual Chassis Mode

Step-by-Step Procedure

To configure (set) the preprovisioned member ID for each MX Series router in the Virtual Chassis, use the **request virtual-chassis member-id set** command. Assigning the member ID causes the router to reboot in preparation for forming the Virtual Chassis.

NOTE: If you issue the **request virtual-chassis member-id set** command without first installing an MX Virtual Chassis Redundancy Feature Pack license on both member routers, the software displays a warning message that you are operating without a valid Virtual Chassis software license.

To configure the member ID and reboot each router to enable Virtual Chassis mode:

1. Log in to the console on member 0 (**gladius**).

2. Set the member ID on member 0.

```
user@gladius> request virtual-chassis member-id set member 0

This command will enable virtual-chassis mode and reboot the system.
 Continue? [yes,no] yes
```

Issuing the **request virtual-chassis member-id** command causes the router to reboot in preparation for membership in the Virtual Chassis.

After the reboot, all MPCs remain powered off until the Virtual Chassis port connection is configured.

3. Log in to the console on member 1 (**trefoil**).

4. Set the member ID on member 1.

```
user@trefoil> request virtual-chassis member-id set member 1

This command will enable virtual-chassis mode and reboot the system.
 Continue? [yes,no] yes
```

After the reboot, all MPCs remain powered off until the Virtual Chassis port connection is configured.

Results Display the results of the configuration on each router. At this point in the procedure, all line cards are offline, and the routers are each designated with the **Master** role because they are not yet interconnected as a fully formed Virtual Chassis. In addition, member 1 (**trefoil**) remains in Amnesiac state (has no defined configuration) until the Virtual Chassis forms and the configuration is committed.

For member 0 (**gladius**):

```
{master:member0-re0}

user@gladius> show virtual-chassis status
Preprovisioned Virtual Chassis
Virtual Chassis ID: 4f2b.1aa0.de08
                                                Mastership           Neighbor List

Member ID       Status   Serial No    Model    priority    Role      ID  Interface

0 (FPC   0- 11) Prsnt    JN10C7135AFC mx240          129  Master*
```

For member 1 (**trefoil**):

```
Amnesiac (ttyd0)

login: user
Password:
...

{master:member1-re0}

user> show virtual-chassis status
```

```
Virtual Chassis ID: eabf.4e50.91e6
Virtual Chassis Mode: Disabled
                                                  Mastership           Neighbor List
Member ID      Status   Serial No    Model    priority    Role      ID  Interface
1 (FPC  12- 23) Prsnt    JN115D117AFB mx480          128  Master*
```

Configuring Virtual Chassis Ports to Interconnect Member Routers

Step-by-Step Procedure

To interconnect the member routers in an MX Series Virtual Chassis, use the **request virtual-chassis vc-port set** command to configure (set) Virtual Chassis ports on Modular Port Concentrator/Modular Interface Card (MPC/MIC) interfaces.

NOTE: If you issue the **request virtual-chassis vc-port set** command without first installing an MX Virtual Chassis Redundancy Feature Pack license on both member routers, the software displays a warning message that you are operating without a valid Virtual Chassis software license.

To configure Virtual Chassis ports on MPC/MIC interfaces to connect the member routers in the Virtual Chassis:

1. Configure the Virtual Chassis ports on member 0 (**gladius**).
   a. Log in to the console on member 0.
   b. Configure the first Virtual Chassis port that connects to member 1 (**trefoil**).

      ```
      {master:member0-re0}
      user@gladius> request virtual-chassis vc-port set fpc-slot 2 pic-slot 2 port 0
      vc-port successfully set
      ```

      After the Virtual Chassis port is created, it is renamed **vcp-*slot/pic/port*** (for example, **vcp-2/2/0**), and the line card associated with that port comes online. The line cards in the other member router remain offline until the Virtual Chassis forms. Each Virtual Chassis port is dedicated to the task of interconnecting member routers in a Virtual Chassis, and is no longer available for configuration as a standard network port.

   c. When **vcp-2/2/0** is up, configure the second Virtual Chassis port that connects to member 1.

      ```
      {master:member0-re0}
      user@gladius> request virtual-chassis vc-port set fpc-slot 2 pic-slot 3 port 0
      vc-port successfully set
      ```

2. Configure the Virtual Chassis ports on member 1 (**trefoil**).
   a. Log in to the console on member 1.
   b. Configure the first Virtual Chassis port that connects to member 0 (**gladius**).

```
{master:member1-re0}

user@trefoil> request virtual-chassis vc-port set fpc-slot 2 pic-slot 0 port 0
vc-port successfully set
```

c. When **vcp-2/0/0** is up, configure the second Virtual Chassis port that connects to member 0.

```
{master:member1-re0}

user@trefoil> request virtual-chassis vc-port set fpc-slot 5 pic-slot 2 port 0
vc-port successfully set
```

When all of the line cards in all of the member routers are online, and the Virtual Chassis has formed, you can issue Virtual Chassis commands from the terminal window of the master router (**gladius**).

3. Verify that the Virtual Chassis is properly configured and operational.

```
{master:member0-re0}

user@gladius> show virtual-chassis status

{master:member0-re0}

user@gladius> show virtual-chassis vc-port all-members
```

See the Verification section for information about interpreting the output of these commands.

4. Commit the configuration on the master router.

The commit step is required to ensure that the configuration groups and Virtual Chassis configuration are propagated to both members of the Virtual Chassis.

Verification

To confirm that the Virtual Chassis configuration is working properly, perform these tasks:

- Verifying the Member IDs and Roles of the Virtual Chassis Members on page 89
- Verifying the Enhanced IP Network Services Configuration on page 90
- Verifying the Operation of the Virtual Chassis Ports on page 91
- Verifying Neighbor Reachability on page 91

Verifying the Member IDs and Roles of the Virtual Chassis Members

Purpose Verify that the member IDs and roles of the routers belonging to the Virtual Chassis are properly configured.

Action Display the status of the members of the Virtual Chassis configuration:

```
{master:member0-re0}

user@gladius> show virtual-chassis status
Preprovisioned Virtual Chassis
Virtual Chassis ID: a5b6.be0c.9525
                                                Mastership           Neighbor List
Member ID       Status   Serial No    Model    priority    Role      ID  Interface

0 (FPC   0- 11) Prsnt    JN10C7135AFC mx240          129  Master*    1  vcp-2/2/0
                                                                     1  vcp-2/3/0

1 (FPC  12- 23) Prsnt    JN115D117AFB mx480          129  Backup     0  vcp-2/0/0
                                                                     0  vcp-5/2/0
```

Meaning The value **Prsnt** in the **Status** column of the output confirms that the member routers specified in the preprovisioned configuration are currently connected to the Virtual Chassis. The display shows that member 0 (**gladius**) and member 1 (**trefoil**), which were both configured with the **routing-engine** role, are functioning as the master router and backup router of the Virtual Chassis, respectively. The **Neighbor List** displays the interconnections between the member routers by means of the Virtual Chassis ports. For example, member 0 is connected to member 1 through **vcp-2/2/0** and **vcp-2/3/0**. The asterisk (*) following **Master** denotes the router on which the command was issued. The **Mastership priority** value is assigned by the software and is not configurable in the current release.

Verifying the Enhanced IP Network Services Configuration

Purpose Verify that enhanced IP network services has been properly configured for the Virtual Chassis.

Action Display the setting of the network services configuration for the master Routing Engine in the Virtual Chassis master router (member0-re0), and for the master Routing Engine in the Virtual Chassis backup router (member1-re0).

```
{master:member0-re0}

user@gladius> show chassis network-services
Network Services Mode: Enhanced-IP


{backup:member1-re0}

user@trefoil> show chassis network-services
Network Services Mode: Enhanced-IP
```

Meaning The output of the **show chassis network services** command confirms that enhanced IP network services is properly configured on both member routers in the Virtual Chassis.

Verifying the Operation of the Virtual Chassis Ports

Purpose Verify that the Virtual Chassis ports are properly configured and operational.

Action Display the status of the Virtual Chassis ports for both members of the Virtual Chassis.

```
{master:member0-re0}

user@gladius> show virtual-chassis vc-port all-members
member0:
--------------------------------------------------------------------------
Interface       Type              Trunk  Status       Speed        Neighbor
or                                 ID                (mbps)       ID  Interface
Slot/PIC/Port
2/2/0           Configured          3    Up           10000        1   vcp-2/0/0
2/3/0           Configured          3    Up           10000        1   vcp-5/2/0


member1:
--------------------------------------------------------------------------
Interface       Type              Trunk  Status       Speed        Neighbor
or                                 ID                (mbps)       ID  Interface
Slot/PIC/Port
2/0/0           Configured          3    Up           10000        0   vcp-2/2/0
5/2/0           Configured          3    Up           10000        0   vcp-2/3/0
```

Meaning The output confirms that the Virtual Chassis ports you configured are operational. For each member router, the **Interface or Slot/PIC/Port** column shows the location of the Virtual Chassis ports configured on that router. For example, the Virtual Chassis ports on **member0-re0** (**gladius**) are **vcp-2/2/0** and **vcp-2/3/0**. In the **Trunk ID** column, the value **3** indicates that a trunk has formed; if a trunk is not present, this field displays the value **-1** . In the **Status** column, the value **Up** confirms that the interfaces associated with the Virtual Chassis ports are operational. The **Speed** column displays the speed of the Virtual Chassis port interface. The **Neighbor ID/Interface** column displays the member IDs and Virtual Chassis port interfaces that connect to this router. For example, the connections to member 0 (**gladius**) are through **vcp-2/0/0** and **vcp-5/2/0** on member 1 (**trefoil**).

Verifying Neighbor Reachability

Purpose Verify that each member router in the Virtual Chassis can reach the neighbor routers to which it is connected.

Action Display the neighbor reachability information for both member routers in the Virtual Chassis.

```
{master:member0-re0}

user@gladius> show virtual-chassis active-topology all-members
member0:
--------------------------------------------------------------------------
  Destination ID        Next-hop

  1                    1(vcp-2/2/0.32768)


member1:
--------------------------------------------------------------------------
  Destination ID        Next-hop

  0                    0(vcp-2/0/0.32768)
```

Meaning The output confirms that each member router in the Virtual Chassis has a path to reach the neighbors to which it is connected. For each member router, the **Destination ID** specifies the member ID of the destination (neighbor) router. The **Next-hop** column displays the member ID and Virtual Chassis port interface of the next-hop to which packets for the destination ID are forwarded. For example, the next-hop from member 0 (**gladius**) to member 1 (**trefoil**) is through Virtual Chassis port interface **vcp-2/2/0.32768**.

Related Documentation

- Interchassis Redundancy and Virtual Chassis Overview on page 19
- Virtual Chassis Components Overview on page 24
- Guidelines for Configuring Virtual Chassis Ports on page 134
- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58

Configuring an MX2020 Member Router in an Existing MX Series Virtual Chassis

Starting in Junos OS Release 15.1, in an existing two-member MX Series Virtual Chassis that includes an MX960 router or an MX2010 router, you can replace one or both of these routers with an MX2020 router. If you replace either the MX960 router or MX2010 router with an MX2020 router, you must follow this procedure to ensure that the new Virtual Chassis forms properly and that any configurations you use reflect the correct Flexible PIC Concentrator (FPC) slot numbering for interfaces configured on the MX2020 member router.

Before you begin:

- Ensure that the existing Virtual Chassis is properly configured and operational before replacing one of the member routers with an MX2020 router.

See "Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis" on page 75.

- Make sure you know the member router combinations supported with an MX2020 router.

  See "Interchassis Redundancy and Virtual Chassis Overview" on page 19.

- Make sure you understand the slot count values for MX960, MX2010, and MX2020 member routers, and how the FPC slots are numbered when you configure a Virtual Chassis with an MX2020 router.

  See "Virtual Chassis Components Overview" on page 24.

To configure an MX2020 member router in an existing MX Series Virtual Chassis with either an MX960 router or an MX2010 router:

1. Replace either the MX960 router or MX2010 router with an MX2020 router.

   Install the interface modules in the same physical slots in the MX2020 router as they were when installed in the MX960 or MX2010 router.

2. Upgrade all four Routing Engines in the Virtual Chassis (two Routing Engines in each member router) to the current Junos OS software release.

   NOTE: **All four Routing Engines in the Virtual Chassis must be the same model and running the same Junos OS software release. For a Virtual Chassis that includes an MX2020 router, all four Routing Engines must have at least 16 gigabytes of memory.**

3. Set the member ID and slot count of the MX960 router or MX2010 router.

   To ensure that a Virtual Chassis configuration with an MX2020 router and either an MX960 router or MX2010 router forms properly, you *must* set the ***slot-count*** value of the MX960 router or MX2010 router to 20 to match the ***slot-count*** of the MX2020 router.

   user@hostA> request virtual-chassis member-id set member *member-id* slots-per-chassis *slot-count*

   For example, assume that the MX960 router or MX2010 router is member 1 in the preprovisioned Virtual Chassis configuration. To set the member ID and slot count for member 1:

   user@hostA> **request virtual-chassis member-id set member 1 slots-per-chassis 20**

   ```
   This command will enable virtual-chassis mode and reboot the system.
   Continue? [yes,no] yes
   ```

   The router reboots in preparation for forming the Virtual Chassis. After the reboot, the FPC slots are renumbered and all MPCs remain powered off until the Virtual Chassis port connection is configured.

4. Edit your Junos OS configuration to rename any interfaces configured on member 1 to reflect how the FPC slots are renumbered when the Virtual Chassis includes an MX2020 member router.

   Before you add the MX2020 router to the Virtual Chassis, member 0 uses FPC slot numbers 0 through 11 with no offset, and member 1 uses slot numbers 12 through 23 with an offset of 20. After you set the slot count of the MX960 or MX2010 router to 20 and reboot the MX2020 member router, member 0 uses slot numbers 0 through 19 with no offset, and member 1 uses slot numbers 20 through 39 with an offset of 20.

   For example, in the following partial configuration, ge-0/0/0 is an aggregated Ethernet link configured on FPC slot 0 in member 0, and ge-12/0/0 is an aggregated Ethernet link configured on FPC slot 0 in member 1.

   ```
   [edit]
   interfaces {
     ge-0/0/0 {
       gigether-options {
         802.3ad ae0;
       }
     }
     ge-12/0/0 {
       gigether-options {
         802.3ad ae0;
       }
     }
     ae0 {
       ...
       ...
     }
   }
   ```

   If you replace member 1 with an MX2020 member router, FPC slot 0 in member 1 is renumbered as FPC slot 20 on member 1. As a result, you must edit the configuration to change ge-12/0/0 to ge-20/0/0, as follows:

   ```
   [edit]
   interfaces {
     ge-0/0/0 {
       gigether-options {
         802.3ad ae0;
       }
     }
     ge-20/0/0 {
       gigether-options {
         802.3ad ae0;
       }
     }
     ae0 {
       ...
       ...
     }
   }
   ```

Release History Table

Release	Description
15.1	Starting in Junos OS Release 15.1, in an existing two-member MX Series Virtual Chassis that includes an MX960 router or an MX2010 router, you can replace one or both of these routers with an MX2020 router.

Related Documentation

- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- Configuring Member IDs for a Virtual Chassis on page 72
- Interchassis Redundancy and Virtual Chassis Overview on page 19
- Virtual Chassis Components Overview on page 24

Switching the Global Master and Backup Roles in a Virtual Chassis Configuration

You can change the mastership in an MX Series Virtual Chassis by switching the global roles of the master router and backup router in the Virtual Chassis configuration. When you change the mastership by issuing the **request virtual-chassis routing-engine master switch** administrative command, the current master router in the Virtual Chassis (also known as the Virtual Chassis protocol master, or VC-M) becomes the backup router, and the current backup router (also known as the Virtual Chassis protocol backup, or VC-B) becomes the master router.

Before you begin:

- Make sure the system configuration is synchronized between the master router and the backup router.

  If the configuration between the member routers is not synchronized when you issue the **request virtual-chassis routing-engine master switch** command, the router displays the following error message and rejects the command.

  ```
  Error: mastership switch request NOT honored, backup not ready
  ```

- Make sure the Virtual Chassis is not in a transition state (for example, the backup router is in the process of disconnecting from the Virtual Chassis) when you issue the **request virtual-chassis routing-engine master switch** command.

  If you attempt to issue the **request virtual-chassis routing-engine master switch** command during a transition state, the router does not process the command.

To switch the global master and backup roles:

- Issue the **request virtual-chassis routing-engine master switch** command from the Virtual Chassis master Routing Engine in the Virtual Chassis master router (VC-Mm):

  ```
  {master:member0-re0}

  user@host> request virtual-chassis routing-engine master switch
  Do you want to continue ? [yes,no] (no) yes
  ```

If you attempt to issue the **request virtual-chassis routing-engine master switch** command from the backup router , the router displays the following error message and rejects the command.

```
error: Virtual Chassis member is not the protocol master
```

Issuing the **request virtual-chassis routing-engine master switch** command from the VC-Mm causes the global role transitions listed in Table 15 on page 96.

Table 15: Virtual Chassis Global Role Transitions Before and After Mastership Switchover

Virtual Chassis Role Before Switching Mastership	Virtual Chassis Role After Switching Mastership
Master Routing Engine in Virtual Chassis master router (VC-Mm)	Standby Routing Engine in Virtual Chassis backup router (VC-Bs)
Standby Routing Engine in Virtual Chassis master router (VC-Ms)	Master Routing Engine in Virtual Chassis backup router (VC-Bm)
Master Routing Engine in Virtual Chassis backup router (VC-Bm)	Master Routing Engine in Virtual Chassis master router (VC-Mm)
Standby Routing Engine in Virtual Chassis backup router (VC-Bs)	Standby Routing Engine in Virtual Chassis master router (VC-Ms)

Related Documentation

- Switchover Behavior in an MX Series Virtual Chassis on page 34
- Virtual Chassis Components Overview on page 24
- Global Roles and Local Roles in a Virtual Chassis on page 30
- Mastership Election in a Virtual Chassis on page 32

Deleting Member IDs in a Virtual Chassis Configuration

In most cases, you delete the member ID from a member router or switch as part of the procedure for deleting a Virtual Chassis configuration. When you delete the member ID by using the **request virtual-chassis member-id delete** command, the router or switch reboots and the software disables Virtual Chassis mode on that device. After the reboot, the router or switch is no longer part of the Virtual Chassis and functions as an independent device.

NOTE: If you issue the **request virtual-chassis member-id delete** command without first installing an MX Virtual Chassis Redundancy Feature Pack license on both member routers, the software displays a warning message that you are operating without a valid Virtual Chassis software license.

A software license is not needed to create an EX9200 Virtual Chassis.

To delete the Virtual Chassis member IDs from both member routers or switches and disable Virtual Chassis mode:

1. In the console window on the router or switch configured as **member 0**, delete member ID **0**.

```
{master:member0-re0}

user@host1> request virtual-chassis member-id delete
This command will disable virtual-chassis mode and reboot the system.
 Continue? [yes,no] (no) yes

Updating VC configuration and rebooting system, please wait...

{master:member0-re0}
user@host1>

*** FINAL System shutdown message from root@host1 ***
System going down IMMEDIATELY
```

2. In the console window on the router or switch configured as **member 1**, delete member ID **1**.

```
{master:member1-re0}

user@host2> request virtual-chassis member-id delete
This command will disable virtual-chassis mode and reboot the system.
 Continue? [yes,no] (no) yes

Updating VC configuration and rebooting system, please wait...

{master:member1-re0}
user@host2>

*** FINAL System shutdown message from root@host2 ***
System going down IMMEDIATELY
```

3. (Optional) Confirm that Virtual Chassis mode has been disabled on both member routers or switches.

   For example:

```
user@host1> show virtual-chassis status
error: the virtual-chassis-control subsystem is not running
```

Related Documentation

- Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 110
- *Configuring an EX9200 Virtual Chassis*
- Example: Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 111

Disabling Split Detection in a Virtual Chassis Configuration

If there is a disruption to a Virtual Chassis due to failure of a member device or one or more Virtual Chassis port links, the resulting connectivity loss can cause a split in the Virtual Chassis configuration. Split detection, which is enabled by default in an MX Series and EX9200 Virtual Chassis, identifies the split and minimizes further network disruption.

You can disable split detection by including the **no-split-detection** statement at the **[edit virtual-chassis]** hierarchy level. Disabling split detection can be useful in certain Virtual Chassis configurations.

For example, if the backup device fails in a two-member Virtual Chassis configuration and split detection is enabled (the default behavior), the master device takes a **line-card** role, and the line cards (FPCs) that do not host Virtual Chassis ports go offline. This state effectively isolates the master router or switch and removes it from the Virtual Chassis until connectivity is restored. As a result, routing or switching is halted and the Virtual Chassis configuration is disabled. By contrast, if the backup router or switch fails in a two-member Virtual Chassis configuration and split detection is disabled, the master router or switch retains mastership and maintains all of the Virtual Chassis ports, effectively resulting in a single-member Virtual Chassis consisting of only the master device.

BEST PRACTICE: We recommend that you disable split detection for a two-member Virtual Chassis configuration if you think the backup router or switch is more likely to fail than the Virtual Chassis port interfaces to the backup router or switch. Configuring redundant Virtual Chassis ports on different line cards in each member router or switch reduces the likelihood that all Virtual Chassis port interfaces to the backup router or switch can fail.

To disable split detection:

1. Specify that you want to disable the default detection of splits in the Virtual Chassis.

```
[edit virtual-chassis]
user@host# set no-split-detection
```

2. Commit the configuration.

Disabling split detection causes different results for different types of Virtual Chassis failures. For information, see "Split Detection Behavior in a Virtual Chassis" on page 38.

Related Documentation

- Split Detection Behavior in a Virtual Chassis on page 38
- Global Roles and Local Roles in a Virtual Chassis on page 30
- Switchover Behavior in an MX Series Virtual Chassis on page 34
- Virtual Chassis Components Overview on page 24

Example: Replacing a Routing Engine in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers

If you remove a Routing Engine from a member router in an MX Series Virtual Chassis for upgrade or repair, you must replace it with a new Routing Engine in the empty Routing Engine slot, and install the same Junos OS release on the new Routing Engine that is running on the other Routing Engines in the Virtual Chassis. The Virtual Chassis remains operational during the replacement procedure.

All four Routing Engines (both Routing Engines in the master router and both Routing Engines in the backup router) in the Virtual Chassis must run the same Junos OS release.

BEST PRACTICE: **We recommend that you replace a Routing Engine in an MX Series Virtual Chassis configuration during a maintenance window to minimize the possibility of disruption to subscribers.**

This example describes how to replace a Routing Engine in an MX Series Virtual Chassis configuration consisting of two MX Series routers, each of which has dual Routing Engines installed:

- Requirements on page 99
- Overview and Topology on page 100
- Configuration on page 102
- Verification on page 106

Requirements

This example uses the following software and hardware components:

- Junos OS Release 11.4 and later releases
- One MX240 3D Universal Edge Router with dual Routing Engines

- One MX480 3D Universal Edge Router with dual Routing Engines

NOTE: This configuration example has been tested using the software release listed and is assumed to work on all later releases.

See Table 16 on page 101 for information about the hardware installed in each MX Series router.

BEST PRACTICE: We recommend that you use the **commit synchronize** command to save any configuration changes to the Virtual Chassis.

For an MX Series Virtual Chassis, the **force** option is the default and only behavior when you issue the **commit synchronize** command. Issuing the **commit synchronize** command for an MX Series Virtual Chassis configuration has the same effect as issuing the **commit synchronize force** command.

Overview and Topology

To replace a Routing Engine in an MX Series Virtual Chassis configuration, you must:

1. Remove the Routing Engine that needs repair or upgrade.
2. Return the Routing Engine to Juniper Networks, Inc.
3. Install the new Routing Engine in the empty Routing Engine slot.
4. Modify the Routing Engine factory configuration to enable formation of the Virtual Chassis.
5. Install the same Junos OS release on the new Routing Engine that is running on the other Routing Engines in the Virtual Chassis.
6. Reboot the new Routing Engine to run the Junos OS software release.

Figure 3 on page 101 shows the topology of the MX Series Virtual Chassis configuration used in this example. This example replaces the backup RE-S-2000 Routing Engine in slot 1 of the Virtual Chassis backup router, which is an MX480 router named **trefoil** that is assigned member ID **1**. The backup Routing Engine in slot 1 of **trefoil** is represented in the example as **member1-re1**.

For redundancy, each of the two member routers is configured with two Virtual Chassis ports.

Figure 3: Sample Topology for a Virtual Chassis with Two MX Series Routers

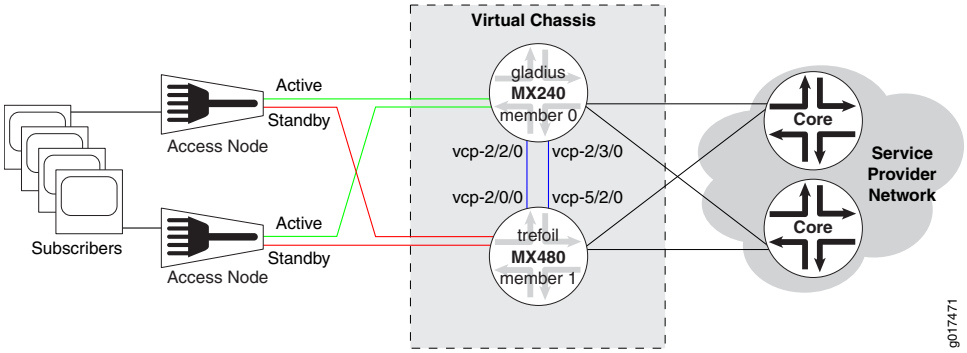


Table 16 on page 101 shows the hardware and software configuration settings for each MX Series router in the Virtual Chassis.

Table 16: Components of the Sample MX Series Virtual Chassis

Router Name	Hardware	Serial Number	Member ID	Role	Virtual Chassis Ports	Network Port Slot Numbering
gladius	MX240 router with: • 60-Gigabit Ethernet Enhanced Queuing MPC • 20-port Gigabit Ethernet MIC with SFP • 4-port 10-Gigabit Ethernet MIC with XFP • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member0-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member0-re1**)	JN10C7135AFC	0	routing-engine (master)	vcp-2/2/0 vcp-2/3/0	FPC 0 – 11

Table 16: Components of the Sample MX Series Virtual Chassis *(continued)*

Router Name	Hardware	Serial Number	Member ID	Role	Virtual Chassis Ports	Network Port Slot Numbering
trefoil	MX480 router with: • Two 30-Gigabit Ethernet Queuing MPCs • Two 20-port Gigabit Ethernet MICs with SFP • Two 2-port 10-Gigabit Ethernet MICs with XFP • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member1-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member1-re1**)	JN115D117AFB	1	routing-engine (backup)	vcp-2/0/0 vcp-5/2/0	FPC 12 – 23 (offset = 12)

Configuration

To replace a Routing Engine in a Virtual Chassis configuration consisting of two MX Series routers, each with dual Routing Engines, perform these tasks:

- Removing the Routing Engine on page 103
- Returning the Routing Engine to Juniper Networks, Inc. on page 103
- Installing the New Routing Engine on page 103
- Modifying the Routing Engine Factory Configuration on page 103
- Installing the Junos OS Release on the New Routing Engine on page 105

Removing the Routing Engine

Step-by-Step Procedure

To remove the Routing Engine that needs repair or upgrade:

- Remove the Routing Engine according to the procedure for your MX Series router.
  - For an MX240 router, see *Removing an MX240 Routing Engine* in the *MX240 3D Universal Edge Router Hardware Guide*.
  - For an MX480 router, see *Removing an MX480 Routing Engine* in the *MX480 3D Universal Edge Router Hardware Guide*.
  - For an MX960 router, see *Removing an MX960 Routing Engine* in the *MX960 3D Universal Edge Router Hardware Guide*.

Returning the Routing Engine to Juniper Networks, Inc.

Step-by-Step Procedure

To return the Routing Engine to Juniper Networks, Inc:

- Obtain a Return Materials Authorization (RMA) from the Juniper Networks Technical Assistance Center (JTAC) and return the Routing Engine to Juniper Networks, Inc.

  For instructions, see *Returning a Hardware Component to Juniper Networks, Inc.* in the *Hardware Guide* for your MX Series router.

Installing the New Routing Engine

Step-by-Step Procedure

To install the new Routing Engine in the Virtual Chassis member router:

- Install the Routing Engine in the empty Routing Engine slot of the member router according to the procedure for your MX Series router.
  - For an MX240 router, see *Installing an MX240 Routing Engine* in the *MX240 3D Universal Edge Router Hardware Guide*.
  - For an MX480 router, see *Installing an MX480 Routing Engine* in the *MX480 3D Universal Edge Router Hardware Guide*.
  - For an MX960 router, see *Installing an MX960 Routing Engine* in the *MX960 3D Universal Edge Router Hardware Guide*.

Modifying the Routing Engine Factory Configuration

Step-by-Step Procedure

A Routing Engine shipped from the factory is loaded with a default factory configuration that includes the following stanza at the [edit] hierarchy level:

```
[edit]
system {
  commit {
    factory-settings {
      reset-virtual-chassis-configuration;
    }
  }
}
```

When this configuration stanza is present, the Routing Engine can operate only in a standalone chassis and *not* in a Virtual Chassis member router. As a result, if you install this Routing Engine in the standby slot of a Virtual Chassis member router (**member1-re1** in this procedure), the Routing Engine does not automatically synchronize with the master Routing Engine and boot in Virtual Chassis mode.

To ensure that the standby factory Routing Engine successfully synchronizes with the master Routing Engine, you must remove the standalone chassis configuration stanza from the standby factory Routing Engine and verify that it reboots in Virtual Chassis mode before you install the Junos OS release.

To modify the Routing Engine factory configuration to ensure proper operation of the Virtual Chassis:

1. Log in to the console of the new Routing Engine as the user **root** with no password.

2. Configure a plain-text password for the **root** (superuser) login.

   ```
   {local:member1-re1}[edit system]
   root# set root-authentication plain-text-password
   New password: type password here
   Retype new password: retype password here
   ```

3. Delete the standalone chassis configuration.

   ```
   {local:member1-re1}[edit]
   root# delete system commit factory-settings reset-virtual-chassis-configuration
   ```

4. Commit the configuration.

   The new Routing Engine synchronizes the Virtual Chassis member ID with the master Routing Engine and boots in Virtual Chassis mode.

5. Verify that the new Routing Engine is in Virtual Chassis mode.

   During the boot process, the router displays the following output to indicate that it has synchronized the Virtual Chassis member ID (1) with the master Routing Engine and is in Virtual Chassis mode.

   ```
   ...
   virtual chassis member-id = 1
   ```



```
virtual chassis mode      = 1
...
```

Installing the Junos OS Release on the New Routing Engine

Step-by-Step Procedure

You must install the same Junos OS release on the new Routing Engine that is running on the other Routing Engines in the MX Series Virtual Chassis. Installing the Junos OS software prepares the Routing Engine to run the new Junos OS release after a reboot. This action is also referred to as *arming* the Routing Engine.

To install the Junos OS release on the new Routing Engine (**member1-re1**) in the Virtual Chassis:

1. Use FTP or a Web browser to download the Junos OS software to the master Routing Engine on the Virtual Chassis master router (**member0-re0**).

   See *Downloading Software* in the *Installation and Upgrade Guide*.

   NOTE: Make sure you download and install the same Junos OS release that is running on all Routing Engines in the Virtual Chassis.

2. If you have not already done so, log in to the console of the new Routing Engine as the user **root** with no password.

3. If you have not already done so, configure a plain-text password for the **root** (superuser) login.

   ```
   {local:member1-re1}[edit system]
   root# set root-authentication plain-text-password
   New password: type password here
   Retype new password: retype password here
   ```

4. Log in to the console of the Virtual Chassis master router (**member0-re0**) as the user **root**.

5. From the console of the Virtual Chassis master router, commit the configuration.

   ```
   {master:member0-re0}[edit]
   root# commit synchronize and-quit
   ...
   member1-re0:
   configuration check succeeds
   member0-re0:
   commit complete
   member1-re0:
   commit complete
   member1-re1:
   commit complete
   ```

```
Exiting configuration mode
```

6. Use Telnet or SSH to log in to the member router containing the new Routing Engine (**trefoil**).

   ```
   {local:member1-re1}
   user@trefoil>
   ```

   Notice that the router name (**trefoil**) now appears in the command prompt.

7. Install the Junos OS release on the new Routing Engine (**member1-re1**) from the Virtual Chassis master router (**member0-re0**).

   ```
   {master:member0-re0}
   user@trefoil> request system software add member member-id re1 no-validate reboot
   package-name force
   ```

   For example:

   ```
   {master:member0-re0}
   user@trefoil> request system software add member 1 re1 no-validate reboot
   /var/tmp/jinstall-11.4R1-8-domestic-signed.tgz force
   Pushing bundle to re1...
   ```

   This command reboots **member1-re1** after the software is added.

Results After the reboot, the new Routing Engine becomes part of the Virtual Chassis, updates its command prompt to display **member1-re1**, and copies the appropriate configuration from the Virtual Chassis.

Verification

To verify that the MX Series Virtual Chassis is operating properly with the new Routing Engine, perform these tasks:

- Verifying the Junos OS Installation on the New Routing Engine on page 106
- Verifying the Junos OS License Installation on the New Routing Engine on page 107
- Switching the Local Mastership in the Member Router to the New Routing Engine on page 108

Verifying the Junos OS Installation on the New Routing Engine

Purpose Verify that you have installed the correct Junos OS release on the new Routing Engine (**member1-re1**).

Action Display the hostname, model name, and version information of the Junos OS release running on the new Routing Engine.

```
{local:member1-re1}

user@trefoil> show version local
Hostname: trefoil
Model: mx480
. . .
JUNOS Base OS boot [11.4R1-8]
JUNOS Base OS Software Suite [11.4R1-8]
. . .
```

Meaning The relevant portion of the **show version local** command output confirms that Junos OS Release 11.4R1-8 was installed as intended.

Verifying the Junos OS License Installation on the New Routing Engine

Purpose Verify that the MX Virtual Chassis Redundancy Feature Pack and the required Junos OS feature licenses are properly installed on the member router containing the new Routing Engine.

For information about license installation, see:

- Installing Junos OS Licenses on Virtual Chassis Member Routers on page 62
- *Software Features That Require Licenses on MX Series Routers Only*

Action Display the Junos OS licenses installed on the new Routing Engine.

```
{local:member1-re1}

user@trefoil> show system license
License usage:
                              Licenses     Licenses    Licenses    Expiry
  Feature name                     used    installed      needed
  subscriber-accounting               0            1           0    permanent
  subscriber-authentication           0            1           0    permanent
  subscriber-address-assignment       0            1           0    permanent
  subscriber-vlan                     0            1           0    permanent
  subscriber-ip                       0            1           0    permanent
  scale-subscriber                    0       256000           0    permanent
  scale-l2tp                          0         1000           0    permanent
  scale-mobile-ip                     0         1000           0    permanent
  virtual-chassis                     0            1           0    permanent
```

Meaning The **show system license** command output confirms that the MX Virtual Chassis Redundancy Feature Pack has been installed on this member router. In addition, the necessary Junos OS feature licenses have been installed to enable use of a particular software feature or scaling level.

Switching the Local Mastership in the Member Router to the New Routing Engine

Purpose Verify that the MX Series Virtual Chassis is operating properly with the new Routing Engine by confirming that the new Routing Engine can take over local mastership from the existing Routing Engine in the Virtual Chassis backup router, **trefoil** (member 1).

Action Switch the local mastership of the Routing Engines in **trefoil** from the Routing Engine in slot 0 (**member1-re0**) to the newly installed Routing Engine in slot 1 (**member1-re1**).

```
{backup:member1-re0}
user@trefoil> request chassis routing-engine master switch
```

Wait approximately 1 minute to display the status and roles of the member routers in the Virtual Chassis after the local switchover.

```
{backup:member1-re1}
user@trefoil> show virtual-chassis status
Preprovisioned Virtual Chassis
Virtual Chassis ID: a5b6.be0c.9525
                                                Mastership           Neighbor List
Member ID       Status   Serial No    Model    priority    Role      ID  Interface

0 (FPC   0- 11) Prsnt    JN10C7135AFC mx240          129  Master     1  vcp-2/2/0
                                                                     1  vcp-2/3/0

1 (FPC  12- 23) Prsnt    JN115D117AFB mx480          129  Backup*    0  vcp-2/0/0
                                                                     0  vcp-5/2/0
```

Meaning Issuing the **request chassis routing-engine master switch** command to initiate the local switchover of the Routing Engines in the Virtual Chassis backup router (**trefoil**) affects only the roles of the Routing Engines in that member router (**member1-re0** and **member1-re1**), but does not change the global mastership of the Virtual Chassis. The output of the **show virtual-chassis status** command confirms that after the local switchover, member 0 (**gladius**) is still the Virtual Chassis master router, and member 1 (**trefoil**) is still the Virtual Chassis backup router.

Before the local switchover, **member1-re0** was the master Routing Engine in the Virtual Chassis backup router (VC-Bm), and **member1-re1** (the new Routing Engine) was the standby Routing Engine in the Virtual Chassis backup router (VC-Bs).

After the local switchover, **member1-re0** and **member1-re1** switch roles. The new Routing Engine, **member1-re1**, becomes the master Routing Engine in the Virtual Chassis backup router (VC-Bm), and **member1-re0** becomes the standby Routing Engine in the Virtual Chassis backup router (VC-Bs).

Table 17 on page 109 lists the role transitions that occur for each member router and Routing Engine before and after the local switchover of the Routing Engines in **trefoil**.

NOTE: The role transitions described in Table 17 on page 109 apply only when you initiate the local switchover from the Virtual Chassis backup router (VC-B). For information about the role transitions that occur when you initiate the local switchover from the Virtual Chassis master router (VC-M), see "Switchover Behavior in an MX Series Virtual Chassis" on page 34.

Table 17: Virtual Chassis Role Transitions Before and After Local Routing Engine Switchover

Virtual Chassis Component	Role *Before* Local Switchover	Role *After* Local Switchover
gladius (member 0)	Virtual Chassis master router (VC-M)	Virtual Chassis master router (VC-M)
trefoil (member 1)	Virtual Chassis backup router (VC-B)	Virtual Chassis backup router (VC-B)
member0-re0	Master Routing Engine in the Virtual Chassis master router (VC-Mm)	Master Routing Engine in the Virtual Chassis master router (VC-Mm)
member0-re1	Standby Routing Engine in the Virtual Chassis master router (VC-Ms)	Standby Routing Engine in the Virtual Chassis master router (VC-Ms)
member1-re0	Master Routing Engine in the Virtual Chassis backup router (VC-Bm)	Standby Routing Engine in the Virtual Chassis backup router (VC-Bs)
member1-re1 (new Routing Engine)	Standby Routing Engine in the Virtual Chassis backup router (VC-Bs)	Master Routing Engine in the Virtual Chassis backup router (VC-Bm)

BEST PRACTICE: After you switch the local mastership of the Routing Engines, full synchronization of the Routing Engines takes approximately 30 minutes to complete. To prevent possible loss of subscriber state information due to incomplete synchronization, we recommend that you wait at least 30 minutes before performing another local switchover, global switchover, or graceful Routing Engine switchover in an MX Series Virtual Chassis configuration.

Related Documentation

- Interchassis Redundancy and Virtual Chassis Overview on page 19
- Virtual Chassis Components Overview on page 24
- Installing Junos OS Licenses on Virtual Chassis Member Routers on page 62

- Switchover Behavior in an MX Series Virtual Chassis on page 34
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- *Installation and Upgrade Guide*

Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers

You can delete an MX Series Virtual Chassis configuration at any time. You might want to do so if your network configuration changes, or if you want to replace one or both MX Series member routers with different MX Series routers.

To delete a Virtual Chassis configuration for MX Series routers:

1. Delete the Virtual Chassis ports from each member router.

   See "Deleting Virtual Chassis Ports in a Virtual Chassis Configuration" on page 138.

2. Delete the definitions and applications for the following configuration groups on each member router:
   - **member0-re0**
   - **member0-re1**
   - **member1-re0**
   - **member1-re1**
3. Delete the preprovisioned member information configured at the **[edit virtual-chassis]** hierarchy level on the master router.
4. Delete any interfaces that were configured on the member routers when the Virtual Chassis was created.
5. Delete the Virtual Chassis member IDs to reboot each router and disable Virtual Chassis mode.

   See "Deleting Member IDs in a Virtual Chassis Configuration" on page 97.

NOTE: You cannot override a Virtual Chassis configuration simply by using the **load override** command to load a different configuration on the router from an ASCII file or from terminal input, as you can with other configurations. The member ID and Virtual Chassis port definitions are not stored in the configuration file, and are still defined even after the new configuration file is loaded.

Related Documentation

- Example: Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 111
- Interchassis Redundancy and Virtual Chassis Overview on page 19
- Virtual Chassis Components Overview on page 24

Example: Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers

You can delete an MX Series Virtual Chassis configuration at any time. You might want to do so if your network configuration changes, or if you want to replace one or both MX Series member routers in the Virtual Chassis with different MX Series routers. After you delete the Virtual Chassis configuration, the routers that were formerly members of the Virtual Chassis function as two independent routers.

This example describes how to delete a Virtual Chassis configuration consisting of two MX Series routers:

- Requirements on page 111
- Overview and Topology on page 112
- Configuration on page 114
- Verification on page 121

Requirements

This example uses the following software and hardware components:

- Junos OS Release 11.2 and later releases
- One MX240 3D Universal Edge Router with dual Routing Engines
- One MX480 3D Universal Edge Router with dual Routing Engines

NOTE: This configuration example has been tested using the software release listed and is assumed to work on all later releases.

See Table 18 on page 113 for information about the hardware installed in each MX Series router.

BEST PRACTICE: We recommend that you use the **commit synchronize** command to save any configuration changes to the Virtual Chassis.

For an MX Series Virtual Chassis, the **force** option is the default and only behavior when you issue the **commit synchronize** command. Issuing the **commit synchronize** command for an MX Series Virtual Chassis configuration has the same effect as issuing the **commit synchronize force** command.

Overview and Topology

To delete an MX Series Virtual Chassis configuration, you must:

1. Delete all Virtual Chassis ports.
2. Remove the definitions and applications of the Virtual Chassis configuration groups.
3. Delete the preprovisioned member information configured at the **[edit virtual-chassis]** hierarchy level.
4. Delete any configured interfaces.
5. Remove the member IDs of each member router.

After you issue the **request virtual-chassis member-id delete** command on each router to remove the member ID, the router reboots and the software disables Virtual Chassis mode on that router.

Because the entire Virtual Chassis configuration is propagated from the master router to the other member router when the Virtual Chassis forms, you must delete each component of the Virtual Chassis configuration from both member routers, even though the component was originally configured only on the master router. For example, even though the preprovisioned member information was configured at the **[edit virtual-chassis]** hierarchy level only on the master router, you must delete the **virtual-chassis** stanza from the other member router in the Virtual Chassis.

NOTE: When deleting the Virtual Chassis, you must also delete all Virtual Chassis-related configuration details from all stanzas, otherwise errors will result upon commit.

NOTE: You cannot override a Virtual Chassis configuration simply by using the load override command to load a different configuration on the router from an ASCII file or from terminal input, as you can with other configurations. The member ID and Virtual Chassis port definitions are not stored in the configuration file, and are still defined even after the new configuration file is loaded.

This example deletes the Virtual Chassis configuration that uses the basic topology shown in Figure 4 on page 113. For redundancy, each member router is configured with two Virtual Chassis ports, both of which must be removed as part of the deletion process.

Figure 4: Sample Topology for a Virtual Chassis with Two MX Series Routers

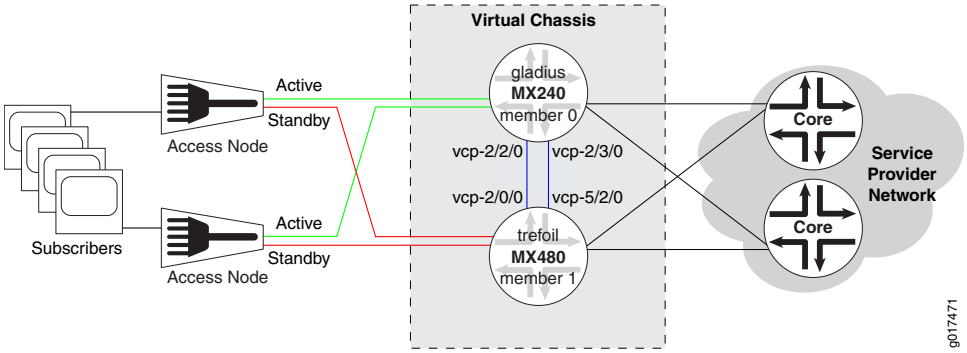


Table 18 on page 113 shows the hardware and software configuration settings for each MX Series router in the Virtual Chassis.

Table 18: Components of the Sample MX Series Virtual Chassis

Router Name	Hardware	Serial Number	Member ID	Role	Virtual Chassis Ports	Network Port Slot Numbering
gladius	MX240 router with: • 60-Gigabit Ethernet Enhanced Queuing MPC • 20-port Gigabit Ethernet MIC with SFP • 4-port 10-Gigabit Ethernet MIC with XFP • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member0-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member0-re1**)	JN10C7135AFC	0	routing-engine (master)	vcp-2/2/0 vcp-2/3/0	FPC 0 – 11

Table 18: Components of the Sample MX Series Virtual Chassis *(continued)*

Router Name	Hardware	Serial Number	Member ID	Role	Virtual Chassis Ports	Network Port Slot Numbering
trefoil	MX480 router with: • Two 30-Gigabit Ethernet Queuing MPCs • Two 20-port Gigabit Ethernet MICs with SFP • Two 2-port 10-Gigabit Ethernet MICs with XFP • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member1-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member1-re1**)	JN115D117AFB	1	routing-engine (backup)	vcp-2/0/0 vcp-5/2/0	FPC 12 – 23 (offset = 12)

Configuration

To delete a Virtual Chassis configuration consisting of two MX Series routers, perform these tasks:

- Deleting Virtual Chassis Ports on page 115
- Deleting Configuration Group Definitions and Applications on page 116
- Deleting Preprovisioned Member Information on page 118
- Deleting Configured Interfaces on page 119
- Deleting Member IDs to Disable Virtual Chassis Mode on page 120

Deleting Virtual Chassis Ports

Step-by-Step Procedure

To delete a Virtual Chassis port from a member router, you must use the **request virtual-chassis vc-port delete** command.

NOTE: If you issue the **request virtual-chassis vc-port delete** command without first installing an MX Virtual Chassis Redundancy Feature Pack license on both member routers, the software displays a warning message that you are operating without a valid Virtual Chassis software license.

To remove the Virtual Chassis ports from each member router:

1. In the console window on member 0 (**gladius**), remove both Virtual Chassis ports (**vcp-2/2/0** and **vcp-2/3/0**).

```
{master:member0-re0}

user@gladius> request virtual-chassis vc-port delete fpc-slot 2 pic-slot 2 port 0
vc-port successfully deleted

{master:member0-re0}

user@gladius> request virtual-chassis vc-port delete fpc-slot 2 pic-slot 3 port 0
vc-port successfully deleted
```

2. In the console window on member 1 (**trefoil**), remove both Virtual Chassis ports (**vcp-2/0/0** and **vcp-5/2/0**).

```
{backup:member1-re0}

user@trefoil> request virtual-chassis vc-port delete fpc-slot 2 pic-slot 0 port 0
vc-port successfully deleted

{backup:member1-re0}

user@trefoil> request virtual-chassis vc-port delete fpc-slot 5 pic-slot 2 port 0
vc-port successfully deleted
```

Results

Display the results of the Virtual Chassis port deletion on each router. Confirm that no Virtual Chassis ports are listed in the output of either the **show virtual-chassis status** command or the **show virtual-chassis vc-port** command.

```
{master:member0-re0}

user@gladius> show virtual-chassis status
Preprovisioned Virtual Chassis
Virtual Chassis ID: 4d6f.54cd.d2c1

                                                Mastership            Neighbor List

Member ID       Status   Serial No    Model    priority    Role      ID  Interface

0 (FPC   0- 11) Prsnt    JN10C7135AFC mx240         129  Master*
```

```
1 (FPC  12- 23) NotPrsnt JN115D117AFB mx480


{master:member0-re0}
user@gladius> show virtual-chassis vc-port
member0:
--------------------------------------------------------------------------
```

TIP: Deleting and then re-creating a Virtual Chassis port in an MX Series Virtual Chassis configuration may cause the Virtual Chassis port to appear as **Absent** in the **Status** column of the **show virtual-chassis vc-port** command display. To resolve this issue, reboot the FPC that hosts the re-created Virtual Chassis port.

Deleting Configuration Group Definitions and Applications

Step-by-Step Procedure

As part of deleting a Virtual Chassis configuration for MX Series routers with dual Routing Engines, you must delete the definitions and applications for the following configuration groups on both member routers:

- **member0-re0**
- **member0-re1**
- **member1-re0**
- **member1-re1**

To retain the information in these configuration groups before you delete them, you must copy them to the standard **re0** and **re1** configuration groups on the router, as described in the following procedure. For example, copy configuration groups **member0-re0** and **member1-re0** to **re0**, and copy **member0-re1** and **member1-re1** to **re1**.

NOTE: The **member*n*-re*n*** naming format for configuration groups is reserved for exclusive use by member routers in MX Series Virtual Chassis configurations.

To delete the configuration group definitions and applications for an MX Series Virtual Chassis:

1. In the console window on member 0 (**gladius**), delete the Virtual Chassis configuration group definitions and applications.

a. Copy the Virtual Chassis configuration groups to the standard configuration groups **re0** and **re1**.

```
{master:member0-re0}[edit]
user@gladius# copy groups member0-re0 to re0
user@gladius# copy groups member0-re1 to re1
```

b. Apply the **re0** and **re1** configuration groups.

```
{master:member0-re0}[edit]
user@gladius# set apply-groups re0
user@gladius# set apply-groups re1
```

c. Delete the Virtual Chassis configuration group definitions.

```
{master:member0-re0}[edit]
user@gladius# delete groups member0-re0
user@gladius# delete groups member0-re1
user@gladius# delete groups member1-re0
user@gladius# delete groups member1-re1
```

d. Delete the Virtual Chassis configuration group applications.

```
{master:member0-re0}[edit]
user@gladius# delete apply-groups member0-re0
user@gladius# delete apply-groups member0-re1
user@gladius# delete apply-groups member1-re0
user@gladius# delete apply-groups member1-re1
```

2. In the console window on member 1 (**trefoil**), delete the Virtual Chassis configuration group definitions and applications.

a. Copy the Virtual Chassis configuration groups to the standard configuration groups **re0** and **re1**.

```
{backup:member1-re0}[edit]
user@trefoil# copy groups member1-re0 to re0
user@trefoil# copy groups member1-re1 to re1
```

b. Apply the **re0** and **re1** configuration groups.

```
{backup:member1-re0}[edit]
user@trefoil# set apply-groups re0
user@trefoil# set apply-groups re1
```

c. Delete the Virtual Chassis configuration group definitions.

```
{backup:member1-re0}[edit]
user@trefoil# delete groups member0-re0
user@trefoil# delete groups member0-re1
user@trefoil# delete groups member1-re0
user@trefoil# delete groups member1-re1
```

d. Delete the Virtual Chassis configuration group applications.

```
{backup:member1-re0}[edit]
user@trefoil# delete apply-groups member0-re0
user@trefoil# delete apply-groups member0-re1
user@trefoil# delete apply-groups member1-re0
user@trefoil# delete apply-groups member1-re1
```

Results Display the results of the configuration. Confirm that configuration groups **member0-re0**, **member 0-re1**, **member1-re0**, and **member1-re1** do not appear in the output of either the **show groups** command or the **show apply-groups** command.

```
[edit]
user@gladius# show groups ?

Possible completions:
  <[Enter]>            Execute this command
  <group_name>         Group name
  global               Group name
  re0                  Group name
  re1                  Group name
  |                    Pipe through a command


[edit]
user@gladius# show apply-groups
  ## Last changed: 2010-12-01 09:17:27 PST
  apply-groups [ global re0 re1 ];
```

Deleting Preprovisioned Member Information

Step-by-Step Procedure You must delete the preprovisioned member information, which was configured at the **[edit virtual-chassis]** hierararchy level on the master router and then propagated to the backup router during the formation of the Virtual Chassis.

To delete the preprovisioned member information for the Virtual Chassis:

1. Delete the **virtual-chassis** configuration stanza on member 0 (**gladius**).

   ```
   {master:member0-re0}[edit]
   user@gladius# delete virtual-chassis
   ```

2. Delete the **virtual-chassis** configuration stanza on member 1 (**trefoil**).

   ```
   {backup:member1-re0}[edit]
   user@trefoil# delete virtual-chassis
   ```

Results Display the results of the deletion. Confirm that the **virtual-chassis** stanza no longer exists on either member router. For example, on **gladius** (member 0):

```
{master:member0-re0}[edit]
user@gladius# show virtual-chassis
<no output>
```

Deleting Configured Interfaces

Step-by-Step Procedure

As part of deleting the Virtual Chassis, we recommend that you delete any interfaces that were configured when the Virtual Chassis was formed. This action ensures that nonexistent interfaces or interfaces belonging to the other member router do not remain on the router after Virtual Chassis mode is disabled.

To delete any interfaces that you configured when creating the Virtual Chassis:

1. In the console window on member 0 (**gladius**), delete any configured interfaces and commit the configuration.

   a. Delete the configured interfaces.

      ```
      {master:member0-re0}[edit]
      user@gladius# delete interfaces
      ```

   b. Commit the configuration on member 0.

      ```
      {master:member0-re0}[edit system]
      user@gladius# commit synchronize
        member0-re0:
        configuration check succeeds
        member0-re1:
        commit complete
        member0-re0:
        commit complete
      ```

2. In the console window on member 1 (**trefoil**), delete any configured interfaces and commit the configuration.

   a. Delete the configured interfaces.

      ```
      {backup:member1-re0}[edit]
      user@trefoil# delete interfaces
      ```

   b. Commit the configuration on member 1.

      ```
      {backup:member1-re0}[edit system]
      user@trefoil# commit synchronize
        member1-re0:
        configuration check succeeds
        member1-re1:
        commit complete
        member1-re0:
        commit complete
      ```

Deleting Member IDs to Disable Virtual Chassis Mode

Step-by-Step Procedure

To delete a member ID from a Virtual Chassis member router, you must use the **request virtual-chassis member-id delete** command.

NOTE: If you issue the **request virtual-chassis member-id delete** command without first installing an MX Virtual Chassis Redundancy Feature Pack license on both member routers, the software displays a warning message that you are operating without a valid Virtual Chassis software license.

To delete the Virtual Chassis member IDs and disable Virtual Chassis mode:

1. In the console window on member 0 (**gladius**), delete the member ID and reboot the router.

   a. Exit configuration mode.

      ```
      {master:member0-re0}[edit]
      user@gladius# exit
      Exiting configuration mode
      ```

   b. Delete member ID **0**.

      ```
      {master:member0-re0}

      user@gladius> request virtual-chassis member-id delete
      This command will disable virtual-chassis mode and reboot the system.
       Continue? [yes,no] (no) yes

      Updating VC configuration and rebooting system, please wait...

      {master:member0-re0}
      user@gladius>

      *** FINAL System shutdown message from root@gladius ***

      System going down IMMEDIATELY
      ```

2. In the console window on member 1 (**trefoil**), delete the member ID and reboot the router.

   a. Exit configuration mode.

      ```
      {master:member1-re0}[edit]
      user@trefoil# exit
      Exiting configuration mode
      ```

   b. Delete member ID **1**.

      ```
      {master:member1-re0}

      user@trefoil> request virtual-chassis member-id delete
      This command will disable virtual-chassis mode and reboot the system.
       Continue? [yes,no] (no) yes
      ```

```
Updating VC configuration and rebooting system, please wait...

{backup:member1-re0}
user@trefoil>

*** FINAL System shutdown message from root@trefoil ***

System going down IMMEDIATELY
```

Results After you issue the **request virtual-chassis member-id delete** command to remove the member ID, the router reboots and the software disables Virtual Chassis mode on that router. The routers that were formerly members of the Virtual Chassis now function as two independent routers.

Display the results of the configuration to confirm that the Virtual Chassis configuration has been deleted on each router. For example, on **gladius** (formerly member 0):

```
user@gladius> show virtual-chassis status
error: the virtual-chassis-control subsystem is not running


user@gladius> show virtual-chassis vc-port
error: the virtual-chassis-control subsystem is not running
```

Verification

To confirm that the Virtual Chassis configuration has been properly deleted, perform these tasks:

- Verifying Deletion of the Virtual Chassis Ports on page 121
- Verifying Deletion of the Virtual Chassis Configuration Groups on page 122
- Verifying Deletion of the Virtual Chassis Member IDs on page 123

Verifying Deletion of the Virtual Chassis Ports

Purpose Verify that the Virtual Chassis ports on both member routers have been deleted from the configuration.

Action Display the status of the Virtual Chassis configuration and Virtual Chassis ports.

```
{master:member0-re0}

user@gladius> show virtual-chassis status
Preprovisioned Virtual Chassis
Virtual Chassis ID: 4d6f.54cd.d2c1

                                                Mastership            Neighbor List

Member ID       Status   Serial No    Model    priority    Role      ID  Interface

0 (FPC   0- 11) Prsnt    JN10C7135AFC mx240         129  Master*
```

```
1 (FPC  12- 23) NotPrsnt JN115D117AFB mx480



{master:member0-re0}
user@gladius> show virtual-chassis vc-port
member0:
--------------------------------------------------------------------------
```

Meaning In the output of the **show virtual-chassis status** command, no Virtual Chassis ports (**vcp-*slot/pic/port***) are displayed in the Neighbor List. The asterisk (*) following **Master** denotes the router on which the **show virtual-chassis status** command command was issued.

In the output of the **show virtual-chassis vc-port** command, no Virtual Chassis ports are displayed on the router on which the command was issued.

Verifying Deletion of the Virtual Chassis Configuration Groups

Purpose Verify that the definitions and applications of the following Virtual Chassis configuration groups have been deleted from the global configuration:

- **member0-re0**
- **member0-re1**
- **member1-re0**
- **member1-re1**

Action Display the status of the Virtual Chassis configuration group definitions and applications.

```
[edit]
user@gladius# show groups ?

Possible completions:
  <[Enter]>            Execute this command
  <group_name>         Group name
  global               Group name
  re0                  Group name
  re1                  Group name
  |                    Pipe through a command


[edit]
user@gladius# show apply-groups
  apply-groups [ global re0 re1 ];
```

Meaning The output confirms that the Virtual Chassis configuration group definitions and applications have been deleted. In the output of both **show groups** and **show apply-groups**, only the standard configuration groups (**global**, **re0**, and **re1**) are listed. The Virtual Chassis

configuration groups (**member0-re0**, **member 0-re1**, **member1-re0**, and **member1-re1**) do not appear.

Verifying Deletion of the Virtual Chassis Member IDs

Purpose Verify that the member IDs for the Virtual Chassis have been deleted, and that the Virtual Chassis is no longer configured on either MX Series router.

Action Display the results of the configuration on each router. For example, on **trefoil** (formerly member 1):

```
user@trefoil> show virtual-chassis status
error: the virtual-chassis-control subsystem is not running


user@trefoil> show virtual-chassis vc-port
error: the virtual-chassis-control subsystem is not running
```

Meaning When you attempt to issue either the **show virtual-chassis status** command or the **show virtual-chassis vc-port** command after the Virtual Chassis has been deleted, the router displays an error message indicating that the Virtual Chassis is no longer configured, and rejects the command.

Related Documentation

- Interchassis Redundancy and Virtual Chassis Overview on page 19
- Virtual Chassis Components Overview on page 24
- Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 110

Upgrading an MX Virtual Chassis SCB or SCBE to SCBE2

To upgrade an MX Virtual Chassis SCB or SCBE to SCBE2, perform the following steps:

NOTE: **SCBE2 does not support smooth upgrade.**

1. Preparing for the SCBE2 Upgrade on page 124
2. Powering Off the MX Series Router on page 124
3. Removing an MX Series Routing Engine from an SCB or SCBE on page 125
4. Replacing the SCB or SCBE with SCBE2 on page 125
5. Installing the MX Series Routing Engine into an SCBE2 on page 126
6. Powering On the MX Series Router on page 126
7. Configuring Member IDs for the Virtual Chassis on page 127

8. Configuring Virtual Chassis Ports on page 129
9. Completing the SCBE2 Upgrade on page 130

Preparing for the SCBE2 Upgrade

To prepare for the SCBE2 upgrade:

1. Verify that the system is runing Junos OS Release 13.3 or later by issuing the **show version** command on the master router.

```
user@host> show version
Junos Base OS Software Suite [13.3-yyyymmdd];
...
```

NOTE: The SCBE2 is supported only on:

- Junos OS Release 13.3 or later
- Network Services Mode: Enhanced-IP

The latest software ensures a healthy system—that is, a system that comprises Routing Engines, control boards, and FPCs—before the upgrade.

For information about how to verify and upgrade Junos OS on MX Virtual Chassis configurations, see "Example: Upgrading Junos OS in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers by Rebooting the Routing Engines" on page 165.

2. Record the local IP address of each RE in the Virtual Chassis configuration. This information will be necessary to reconfigure the VC member IDs after the upgrade.

3. Apply a system halt before shutting down the system by issuing the **request system halt all-members** command on the master router.

```
user@host> request system halt all-members
```

Powering Off the MX Series Router

NOTE: After turning off the power supply, wait at least 60 seconds before turning it back on.

To power off the MX Series router:

1. On the external management device connected to the Routing Engine, issue the **request system halt all-members** operational mode command. The command shuts down the Routing Engines cleanly, so that their state information is preserved.

   ```
   user@host> request system halt all-members
   ```

2. Wait until a message appears on the console confirming that the operating system has halted.

3. Attach an electrostatic discharge (ESD) grounding strap to your bare wrist and connect the strap to one of the ESD points on the chassis.

4. Move the AC input switch on the chassis above the AC power supply or the DC circuit breaker on each DC power supply faceplate to the off **(O)** position.

Removing an MX Series Routing Engine from an SCB or SCBE

To remove an MX Series Routing Engine from an SCB or SCBE:

Remove the Routing Engine according to the procedure for your MX Series router.

- For an MX240 router, see *Removing an MX240 Routing Engine* in the *MX240 3D Universal Edge Router Hardware Guide*.
- For an MX480 router, see *Removing an MX480 Routing Engine* in the *MX480 3D Universal Edge Router Hardware Guide*.
- For an MX960 router, see *Removing an MX960 Routing Engine* in the *MX960 3D Universal Edge Router Hardware Guide*.

Replacing the SCB or SCBE with SCBE2

To replace the existing SCB or SCBE with SCBE2:

1. Attach an electrostatic discharge (ESD) grounding strap to your bare wrist and connect the strap to one of the ESD points on the chassis.

2. Remove and replace the offline SCB or SCBE on the router with SCBE2.

Installing the MX Series Routing Engine into an SCBE2

After removing the routing engine from the original SCB or SCBE, you can install it into the new SCBE2. Follow these instructions to install a MX Series Routing Engine into an SCBE2:

1. Attach an electrostatic discharge (ESD) grounding strap to your bare wrist, and connect the strap to one of the ESD points on the chassis.
2. Ensure that the ejector handles are not in the locked position. If necessary, flip the ejector handles outward.
3. Place one hand underneath the Routing Engine to support it.
4. Carefully align the sides of the Routing Engine with the guides inside the opening on the SCBE2.
5. Slide the Routing Engine into the SCBE2 until you feel resistance and then press the faceplate of the Routing Engine until it engages the connectors.
6. Press both of the ejector handles inward to seat the Routing Engine.
7. Tighten the captive screws on the top and bottom of the Routing Engine.
8. Connect the management device cables to the Routing Engine.

Powering On the MX Series Router

To power on the MX Series router:

1. Verify that the power supplies are fully inserted in the chassis.
2. Verify that each AC power cord is securely inserted into its appliance inlet.
3. Verify that an external management device is connected to one of the Routing Engine ports (**AUX**, **CONSOLE**, or **ETHERNET**).
4. Turn on the power to the external management device.
5. Switch on the dedicated customer-site circuit breakers. Follow the ESD and safety instructions for your site.
6. Attach an ESD grounding strap to your bare wrist and connect the strap to one of the ESD points on the chassis.

7. Move the AC input switch on the chassis above the AC power supply or the DC circuit breaker on each DC power-supply faceplate to the off **(—)** position.

8. Check that the AC or the DC power supply is correctly installed and functioning normally. Verify that the **AC OK** and **DC OK** LEDs light steadily, and the **PS FAIL** LED is not lit.

   NOTE: After a power supply is powered on, it can take up to 60 seconds for status indicators—such as the status LEDs on the power supply and the show chassis command display—to indicate that the power supply is functioning normally. Ignore error indicators that appear during the first 60 seconds.

   If any of the status LEDs indicates that the power supply is not functioning normally, repeat the installation and cabling procedures.

9. On the external management device connected to the Routing Engine, monitor the startup process to verify that the system has booted properly.

   NOTE: If the system is completely powered off when you power on the power supply, the Routing Engine boots as the power supply completes its startup sequence. Normally, the router boots from the Junos OS on the CompactFlash card.

   After turning on a power supply, wait at least 60 seconds before turning it off.

Configuring Member IDs for the Virtual Chassis

To re-enable the Virtual Chassis, you must configure the member ID on both devices in the Virtual Chassis configuration.

1. Set the member ID on the router configured as member 0:

   user@hostA> request virtual-chassis member-id set **member 0**

   ```
   This command will enable virtual-chassis mode and reboot the system.
   Continue? [yes,no] yes
   ```

2. Repeat Step 1 to set the member ID on the router configured as **member 1**.

   user@hostB> request virtual-chassis member-id set **member 1**

   ```
   This command will enable virtual-chassis mode and reboot the system.
   Continue? [yes,no] yes
   ```

   The router reboots in preparation for forming the Virtual Chassis. After the reboot, all MPCs remain powered off until the Virtual Chassis port connection is configured.

3. (Optional) Verify the member ID configuration for **member 0**.

   For example:

```
{master:member0-re0}

user@hostA> show virtual-chassis status
Preprovisioned Virtual Chassis
Virtual Chassis ID: 4f2b.1aa0.de08
                                                Mastership           Neighbor
List
Member ID       Status   Serial No    Model    priority    Role      ID
Interface

0 (FPC   0- 11) Prsnt    JN10C7135AFC mx240          129  Master*
```

4. (Optional) Verify the member ID configuration for **member 1**.

   For example:

```
Amnesiac (ttyd0)

login: user
Password:
...

{master:member1-re0}

user> show virtual-chassis status
Virtual Chassis ID: ef98.2c6c.f7f7
                                                Mastership           Neighbor
List
Member ID       Status   Serial No    Model    priority    Role      ID
Interface

1 (FPC  12- 23) Prsnt    JN115D117AFB mx480          128  Master*
```

NOTE: At this point in the configuration procedure, all line cards are offline, and the routers are each designated with the Master role because they are not yet interconnected as a fully formed Virtual Chassis. In addition, member 1 remains in Amnesiac state (has no defined configuration) until the Virtual Chassis forms and the configuration is committed.

Configuring Virtual Chassis Ports

Wait until the system is fully booted back up after setting the VC member IDs. VCP ports need to be defined in order for line cards to function properly in the VC configuration.

1. Log into the console port on member 0 and configure the first Virtual Chassis port that connects to member 1.

   ```
   {master:member0-re0}
   user@host> request virtual-chassis vc-port set fpc-slot number pic-slot number port number
   vc-port successfully set
   ```

   After the Virtual Chassis port is created, it is renamed **vcp-*slot/pic/port*** (for example, **vcp-2/2/0**), and the line card associated with that port comes online. The line cards in the other member router remain offline until the Virtual Chassis forms. Each Virtual Chassis port is dedicated to the task of interconnecting member routers in a Virtual Chassis, and is no longer available for configuration as a standard network port.

2. Log into the console port on member 1 and configure the first Virtual Chassis port that connects to member 0.

   ```
   {master:member1-re0}
   user@host> request virtual-chassis vc-port set fpc-slot number pic-slot number port number
   vc-port successfully set
   ```

3. After establishing the VCP link, wait for all line cards and REs to come online. Confirm the availability of ports that will be used as VCP links by issuing the **show interface** command.

   ```
   user@host> show interface xe-0/0/1
   Physical interface: xe-0/0/1, Enabled, Physical link is Up
     Interface index: 49195, SNMP ifIndex: 591
     Link-level type: Ethernet, MTU: 1514, Speed: 10Gbps, Duplex: Full-Duplex,
   BPDU Error: None, MAC-REWRITE Error: None, Loopback: Disabled, Source filtering:
    Disabled,
     Flow control: Disabled
     Device flags   : Present Running
     Interface flags: SNMP-Traps Internal: 0x0
     Link flags     : None
     CoS queues     : 12 supported, 12 maximum usable queues
     Current address: 00:1d:b5:f7:4e:e1, Hardware address: 00:1d:b5:f7:4e:e1
     Last flapped   : 2011-06-01 00:42:03 PDT (00:02:42 ago)
     Input rate     : 0 bps (0 pps)
     Output rate    : 0 bps (0 pps)
     Active alarms  : None
     Active defects : None

     Logical interface xe-0/0/1.0 (Index 73) (SNMP ifIndex 523)
       Flags: SNMP-Traps 0x0 Encapsulation: ENET2
       Input packets : 0
       Output packets: 0
   ```

```
    Protocol eth-switch, MTU: 0
      Flags: Trunk-Mode
```

4. Configure the remaining VCP ports 1 at a time. Wait 30 seconds after setting each VCP port.

```
user@host> request virtual-chassis vc-port set fpc-slot number pic-slot number port number
vc-port successfully set
```

Completing the SCBE2 Upgrade

To complete the SCBE2 upgrade procedure:

1. Verify that the installation is successful and the SCBE2 is online by issuing the **show chassis environment cb** command:

```
user@host> show chassis environment cb 0
CB 0 status
State  Online
Temperature 30 degrees C / 86 degrees F
...
```

```
user@host> show chassis environment cb 1
CB 1 status
State  Online
Temperature 30 degrees C / 86 degrees F
...
```

   Other details, such as, temperature, power, etc are also displayed along with the state.

2. Verify that the fabric planes come online correctly by issuing the **show chassis fabric summary** command:

```
user@host> show chassis fabric summary
Plane   State    Uptime
 0      Online   2 days, 19 hours, 10 minutes, 9 seconds
 1      Online   2 days, 19 hours, 10 minutes, 9 seconds
...
```

3. Verify that the backup Routing Engine is back online by issuing the **show chassis routing-engine 1** command:

```
user@host> show chassis routing-engine 1
Routing Engine Status:
Slot 1:
Current State  Backup
...
```

4. Verify the SCBE2s before you finish by issuing the **show chassis hardware** command:

```
user@host> show chassis hardware
Hardware inventory:
Item             Version  Part number  Serial number     Description
CB 0             REV 08   750-048307   CABC9829          Enhanced MX SCB 2
CB 1             REV 08   750-048307   CABC9828          Enhanced MX SCB 2
...
```

You see that the router now has SCBE2s.

Related Documentation

- Example: Upgrading Junos OS in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers by Rebooting the Routing Engines on page 165
- Configuring Member IDs for a Virtual Chassis on page 72
- Example: Replacing a Routing Engine in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 99

CHAPTER 4

Configuring Virtual Chassis Ports to Interconnect Member Devices

- Guidelines for Configuring Virtual Chassis Ports on page 134
- Configuring Virtual Chassis Ports to Interconnect Member Routers or Switches on page 136
- Deleting Virtual Chassis Ports in a Virtual Chassis Configuration on page 138

Guidelines for Configuring Virtual Chassis Ports

To interconnect the member routers in a Virtual Chassis for MX Series 3D Universal Edge Routers, you must configure Virtual Chassis ports on Modular Port Concentrator/Modular Interface Card (MPC/MIC) interfaces. After it is configured, a Virtual Chassis port is dedicated to the task of interconnecting member routers, and is no longer available for configuration as a standard network port.

NOTE: Starting with Junos OS Release 14.1, you can preconfigure ports that are currently unavailable for use. Although a Virtual Chassis port is unavailable for use as a standard network port, you can configure this port as a standard network port even after you configure it as a Virtual Chassis port. However, the router does not apply the configuration until you delete the Virtual Chassis port from the Virtual Chassis configuration.

Consider the following guidelines when you configure Virtual Chassis ports in an MX Series Virtual Chassis:

- An MX Series Virtual Chassis supports up to 16 Virtual Chassis ports per trunk.

  If two or more Virtual Chassis ports of the same type and speed (that is, either all 10-Gigabit Ethernet Virtual Chassis ports or all 1-Gigabit Ethernet Virtual Chassis ports) are configured between the same two member routers in an MX Series Virtual Chassis, the Virtual Chassis Control Protocol (VCCP) bundles these Virtual Chassis port interfaces into a trunk, reduces the routing cost accordingly, and performs traffic load balancing across all of the Virtual Chassis port interfaces in the trunk.

- An MX Series Virtual Chassis does *not* support a combination of 1-Gigabit Ethernet (**ge** media type) Virtual Chassis ports and 10-Gigabit Ethernet (**xe** media type) Virtual Chassis ports within the same Virtual Chassis.

  You must configure either all 10-Gigabit Virtual Chassis ports or all 1-Gigabit Virtual Chassis ports in the same Virtual Chassis. We recommend that you configure Virtual Chassis ports on 10-Gigabit Ethernet (**xe**) interfaces.

  This restriction has no effect on access ports or uplink ports in an MX Series Virtual Chassis configuration.

- Configure redundant Virtual Chassis ports that reside on different line cards in each member router.

NOTE: Virtual Chassis ports should be spread over all power zones in each chassis to make the best use of physical redundancy in the router. Distributing VCP interfaces minimizes the risk of split-mastership if power zones are de-energized by PEM or power feed loss. With distributed VCP interfaces, survival of a single power zone prevents split-master and other undesirable protocol mastership conditions from occurring until all zones are de-energized.

For a two-member MX Series Virtual Chassis, we recommend that you configure a minimum of two 10-Gigabit Ethernet Virtual Chassis ports on different line cards in each member router, for a total minimum of four 10-Gigabit Ethernet Virtual Chassis ports in the Virtual Chassis. In addition, make sure the Virtual Chassis port bandwidth is equivalent to no less than 50 percent of the aggregate bandwidth required for user data traffic. The following examples illustrate these recommendations:

- If the bandwidth in your network is equivalent to two 10-Gigabit Ethernet interfaces (20 Gbps) on the access-facing side of the Virtual Chassis and two 10-Gigabit Ethernet interfaces (20 Gbps) on the core-facing side of the Virtual Chassis, we recommend that you configure two 10-Gigabit Ethernet Virtual Chassis ports, which is the recommended minimum in a Virtual Chassis for redundancy purposes.
- If the aggregate bandwidth in your network is equivalent to ten 10-Gigabit Ethernet interfaces (100 Gbps), we recommend that you configure a minimum of five 10-Gigabit Ethernet Virtual Chassis ports, which is 50 percent of the aggregate bandwidth.

- A user data packet traversing the Virtual Chassis port interfaces between member routers is discarded at the Virtual Chassis egress port if the MTU size of the packet exceeds 9150 bytes.

  The maximum MTU size of a Gigabit Ethernet interface or 10-Gigabit Ethernet interface on a single MX Series router is 9192 bytes. In an MX Series Virtual Chassis configuration, user data packets that traverse Gigabit Ethernet or 10-Gigabit Ethernet Virtual Chassis port interfaces have 42 extra bytes of Virtual Chassis-specific header data, which reduces their maximum MTU (payload) size to 9150 bytes. The user data packet is transmitted in its entirety across the Virtual Chassis port interface. However, because packet fragmentation and reassembly is not supported on Virtual Chassis port interfaces, user data packets that exceed 9150 bytes are discarded at the Virtual Chassis egress port.

Release History Table

Release	Description
14.1	Starting with Junos OS Release 14.1, you can preconfigure ports that are currently unavailable for use.

Related Documentation

- Virtual Chassis Components Overview on page 24
- Configuring Virtual Chassis Ports to Interconnect Member Routers or Switches on page 136
- Class of Service Overview for Virtual Chassis Ports on page 145
- Guidelines for Configuring Class of Service for Virtual Chassis Ports on page 150
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Configuring Virtual Chassis Ports to Interconnect Member Routers or Switches

To interconnect the member routers in an MX Series Virtual Chassis, you must use the **request virtual-chassis vc-port set** command to configure network ports into Virtual Chassis ports on Modular Port Concentrator/Modular Interface Card (MPC/MIC) interfaces. To interconnect the member switches into an EX9200 Virtual Chassis, you must use the **request virtual-chassis vc-port set** command to configure network ports into Virtual Chassis ports on line card interfaces. After the **request virtual-chassis vc-port set** is configured on both ends of the link, a Virtual Chassis port that is dedicated to the task of interconnecting member devices is created and the link can no longer be used as a standard network port.

NOTE: If you issue the **request virtual-chassis vc-port set** command without first installing an MX Virtual Chassis Redundancy Feature Pack license on both member routers in an MX Series Virtual Chassis, the software displays a warning message that you are operating without a valid Virtual Chassis software license.

A software license is not needed to create an EX9200 Virtual Chassis.

To configure Virtual Chassis ports:

1. Configure the Virtual Chassis ports on the router or switch configured as member 0.

   a. Configure the first Virtual Chassis port that connects to member 1.

      ```
      {local:member0-re0}
      ```

      ```
      user@hostA> request virtual-chassis vc-port set fpc-slot fpc-slot-number pic-slot
      pic-slot-number port port-number
      ```

      After the Virtual Chassis port is created, it is renamed **vcp-*slot/pic/port***, and the line card associated with that port comes online. The line cards in the other member devices remain offline until the Virtual Chassis forms.

      For example, the following command configures Virtual Chassis port **vcp-2/2/0** on member 0:

      ```
      {local:member0-re0}
      ```

      ```
      user@hostA> request virtual-chassis vc-port set fpc-slot 2 pic-slot 2 port 0
      vc-port successfully set
      ```

   b. When the first Virtual Chassis port is up on member 0, repeat Step 1a to configure the second Virtual Chassis port that connects to member 1.

      ```
      {local:member0-re0}
      ```

      ```
      user@hostA> request virtual-chassis vc-port set fpc-slot fpc-slot-number pic-slot
      pic-slot-number port port-number
      ```

2. Configure the Virtual Chassis ports on the device configured as member 1.

a. Configure the first Virtual Chassis port that connects to member 0.

```
{master:member1-re0}
user@hostB> request virtual-chassis vc-port set fpc-slot fpc-slot-number pic-slot
pic-slot-number port port-number
```

b. When the first Virtual Chassis port is up on member 1, repeat Step 2a to configure the second Virtual Chassis port that connects to member 0.

```
{master:member1-re0}
user@hostB> request virtual-chassis vc-port set fpc-slot fpc-slot-number pic-slot
pic-slot-number port port-number
```

When all of the line cards in all of the member routers or switches are online, and the Virtual Chassis has formed, you can issue Virtual Chassis commands from the terminal window of the master router or switch.

NOTE: When the Virtual Chassis forms, the FPC slots are renumbered to reflect the slot numbering and offsets used in the Virtual Chassis instead of the physical slot numbers where the FPC is actually installed. Member 0 in the Virtual Chassis uses FPC slot numbers 0 through 11 with no offset, and member 1 uses FPC slot numbers 12 through 23, with an offset of 12.

For example, a 10-Gigabit Ethernet interface that appears as xe-14/2/2 (FPC slot 14, PIC slot 2, port 2) in the show interfaces command output is actually interface xe-2/2/2 (FPC slot 2, PIC slot 2, port 2) on member 1 after deducting the FPC slot numbering offset of 12 for member 1.

3. (Optional) Verify the that the Virtual Chassis is properly configured and that the Virtual Chassis ports are operational.

```
{master:member0-re0}
user@hostA> show virtual-chassis status

{master:member0-re0}
user@hostA> show virtual-chassis vc-port all-members
```

4. Commit the configuration on the master router or switch.

The commit step is required to ensure that the configuration groups and Virtual Chassis configuration are propagated to both members of the Virtual Chassis.

BEST PRACTICE: We recommend that you use the commit synchronize command to save any configuration changes to the Virtual Chassis.

For an MX Series or Virtual Chassis, the force option is the default and only behavior when you issue the commit synchronize command. Issuing the commit synchronize command for an MX Series Virtual Chassis

configuration has the same effect as issuing the **commit synchronize force** command.

Related Documentation

- Guidelines for Configuring Virtual Chassis Ports on page 134
- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- *Configuring an EX9200 Virtual Chassis*
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Deleting Virtual Chassis Ports in a Virtual Chassis Configuration

You can delete a Virtual Chassis port (**vcp-*slot/pic/port***) as part of the procedure for deleting a Virtual Chassis configuration. You can also delete a Virtual Chassis port when you want to replace it with a Virtual Chassis port configured on a different FPC slot, PIC slot, or port number in the router or switch. After you delete a Virtual Chassis port by using the **request virtual-chassis vc-port delete** command, the port becomes available to the global configuration and can again function as a standard network port.

NOTE: If you issue the **request virtual-chassis vc-port delete** command without first installing an MX Virtual Chassis Redundancy Feature Pack license on both member routers, the software displays a warning message that you are operating without a valid Virtual Chassis software license.

A software license is not needed to create an EX9200 Virtual Chassis.

To remove the Virtual Chassis ports from both member devices in a Virtual Chassis:

1. In the console window on the router or switch configured as **member 0**, remove one or more Virtual Chassis ports.

   ```
   {master:member0-re0}
   ```

   ```
   user@host1> request virtual-chassis vc-port delete fpc-slot fpc-slot-number pic-slot pic-slot-number port port-number
   ```

   For example, the following command deletes **vcp-2/2/0** (the Virtual Chassis port on FPC slot 2, PIC slot 2, and port 0) from **member 0** in the Virtual Chassis.

   ```
   {master:member0-re0}
   ```

   ```
   user@host1> request virtual-chassis vc-port delete fpc-slot 2 pic-slot 2 port 0
   vc-port successfully deleted
   ```

2. In the console window on the router or switch configured as **member 1**, remove one or more Virtual Chassis ports.

```
{master:member1-re0}

user@host2> request virtual-chassis vc-port delete fpc-slot fpc-slot-number pic-slot
pic-slot-number port port-number
```

3. (Optional) Confirm that the Virtual Chassis ports have been deleted from each of the two member routers or switches.

   When you delete a Virtual Chassis port, its name (**vcp-*slot/pic/port***) no longer appears in the output of the **show virtual-chassis vc-port** command. For example, the following output for the **show virtual-chassis vc-port** command on each member router or switch confirms that all Virtual Chassis ports have been deleted from both member devices.

   For member 0 (**host1**):

```
{master:member0-re0}

user@host1> show virtual-chassis vc-port all-members
member0:
--------------------------------------------------------------------------
```

   For member 1 (**host2**):

```
{backup:member1-re0}

user@host2> show virtual-chassis vc-port all-members
member1:
--------------------------------------------------------------------------
```

TIP: Deleting and then re-creating a Virtual Chassis port configuration may cause the Virtual Chassis port to appear as **Absent** in the **Status** column of the **show virtual-chassis vc-port** command display. To resolve this issue, reboot the FPC that hosts the re-created Virtual Chassis port.

Related Documentation

- Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 110
- *Configuring an EX9200 Virtual Chassis*
- Example: Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 111
- Guidelines for Configuring Virtual Chassis Ports on page 134

CHAPTER 5

Configuring Locality Bias to Conserve Bandwidth on Virtual Chassis Ports

- Locality Bias in a Virtual Chassis on page 141
- Guidelines for Configuring Locality Bias in a Virtual Chassis on page 143
- Configuring Locality Bias for a Virtual Chassis on page 144

Locality Bias in a Virtual Chassis

By default, member routers in an MX Series Virtual Chassis distribute egress traffic equally across all egress port links. Starting with Junos OS Release 14.1, if you want to conserve bandwidth across the internal Virtual Chassis ports, you can use locality bias to direct unicast transit traffic to egress links on the same (local) member router.

- How Locality Bias Works on page 141
- Locality Bias Percentages on page 142

How Locality Bias Works

Locality bias conserves Virtual Chassis port bandwidth in a two-member MX Series Virtual Chassis by directing unicast transit traffic for equal-cost multipath (ECMP) groups and aggregated Ethernet bundles to egress links in the same (local) member router, provided that the local member router has an equal or larger number of available egress links than the remote member router. Because locality bias directs all of the traffic towards the local member router, Virtual Chassis ports do not use bandwidth.

However, if the number of available remote member router egress links exceeds the number of available local member router egress links, the system reduces the amount of traffic in the local member router by using a ratio that is based on the number of remote versus local links. The amount of traffic that the system does not direct toward egress links on the local member router is then split evenly across the egress links in the remote member router.

If either the local member router or remote member router do not have available egress links, then the traffic forwarding state across the Virtual Chassis ports does not change.

Locality Bias Percentages

The router uses the following algorithms to determine the percentage of traffic that is directed toward the local member router egress links, where *L* is the number of egress links on the local member router and *R* is the number of egress links on the remote member router.

BEST PRACTICE: To avoid possible traffic loss and oversubscription on egress interfaces, make sure you understand the utilization requirements, such as total and available bandwidth, for the local links in your network before changing the locality bias configuration.

- If *L >= R*, then *Locality Bias Percentage* = 100 percent and the local member router receives all egress traffic.

  For example, if the local member router and remote member router each contain one egress link, then the locality bias is 100 percent. The router directs all unicast transit traffic that is destined for an ECMP group or aggregated Ethernet bundle to the local member router.

- If *L < R*, then *Locality Bias Percentage* = 200 * (L / (R + L))

  For example, if the local member router (*L*) contains one link and the remote member router (*R*) contains two links, the locality bias percentage calculation is

  200 * (1 / (2 + 1)) = 66

  This means that the system directs 66 percent of the unicast transit traffic destined for an ECMP group or aggregated Ethernet bundle toward the local member router. The system splits the remaining 34 percent of the unicast transit traffic equally between the remote member router egress links. Each of the two remote egress links in the example receives 17 percent of the traffic.

  NOTE: The actual amount of traffic that the local member router receives can vary slightly from the percentages in the algorithm calculations.

- If *L = 0* or *R = 0*, then locality bias does not change the forwarding state.

For both the *L < R* and *L >= R* algorithms, locality bias percentages are recalculated on each line card whenever one of the aggregated Ethernet child links goes up or down, or whenever a link is added to or removed from an ECMP bundle.

NOTE: If an ECMP bundle has one or more child links that are aggregated Ethernet links, then those aggregated Ethernet child links are always considered remote unless *all* of the aggregated Ethernet child links are local.

Release History Table

Release	Description
14.1	Starting with Junos OS Release 14.1, if you want to conserve bandwidth across the internal Virtual Chassis ports, you can use locality bias to direct unicast transit traffic to egress links on the same (local) member router.

Related Documentation

- Guidelines for Configuring Locality Bias in a Virtual Chassis on page 143
- Configuring Locality Bias for a Virtual Chassis on page 144

Guidelines for Configuring Locality Bias in a Virtual Chassis

Starting in Junos OS 14.1, you can configure locality bias in an MX Series Virtual Chassis. Consider the following guidelines when doing so:

- Make sure you know the total and available amounts of bandwidth in your network. Enabling locality bias when local egress links do not have enough bandwidth to support additional traffic can result in immediate traffic loss. The system does not prevent configurations that result in oversubscription.
- Locality bias directs only user traffic toward the local member router. Locality bias does not affect load balancing of control traffic within the Virtual Chassis.
- You cannot configure adaptive aggregated Ethernet load balancing and locality bias together.
- You cannot configure weighted load balancing and locality bias together.

Release History Table

Release	Description
14.1	Starting in Junos OS 14.1, you can configure locality bias in an MX Series Virtual Chassis.

Related Documentation

- Locality Bias in a Virtual Chassis on page 141
- *Understanding Aggregated Ethernet Load Balancing*
- Configuring Locality Bias for a Virtual Chassis on page 144

Configuring Locality Bias for a Virtual Chassis

Starting in Junos OS Release 14.1, configuring locality bias enables you to conserve Virtual Chassis port bandwidth, reduce infrastructure costs, and reduce network latency in a two-member MX Series Virtual Chassis. Locality bias works by directing unicast traffic for equal-cost multipath (ECMP) groups and aggregated Ethernet bundles to egress links in the same (local) member router in the Virtual Chassis rather than to egress links in the remote member router.

You can enable locality bias by including the **locality-bias** statement at the **[edit virtual-chassis]** hierarchy level.

BEST PRACTICE: **To avoid possible traffic loss and oversubscription on egress interfaces, make sure you understand the utilization requirements for the local links in your network before changing the locality bias configuration.**

To configure locality bias for an MX Series Virtual Chassis:

1. Specify that you want to enable locality bias in the Virtual Chassis.

   ```
   [edit virtual-chassis]
   user@host# set locality-bias
   ```

2. Commit the configuration.

Release History Table

Release	Description
14.1	Starting in Junos OS Release 14.1, configuring locality bias enables you to conserve Virtual Chassis port bandwidth, reduce infrastructure costs, and reduce network latency in a two-member MX Series Virtual Chassis.

Related Documentation

- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Locality Bias in a Virtual Chassis on page 141
- Guidelines for Configuring Locality Bias in a Virtual Chassis on page 143

CHAPTER 6

Configuring Class of Service for Virtual Chassis Ports

- Class of Service Overview for Virtual Chassis Ports on page 145
- Guidelines for Configuring Class of Service for Virtual Chassis Ports on page 150
- Example: Configuring Class of Service for Virtual Chassis Ports on MX Series 3D Universal Edge Routers on page 150

Class of Service Overview for Virtual Chassis Ports

By default, all Virtual Chassis port interfaces in a Virtual Chassis for MX Series 3D Universal Edge Routers use a default class of service (CoS) configuration specifically tailored for Virtual Chassis ports. The default configuration, which applies to all Virtual Chassis ports in the Virtual Chassis, includes classifiers, forwarding classes, rewrite rules, and schedulers. In most cases, the default CoS configuration is adequate for your needs without requiring any additional CoS configuration.

In some cases, however, you might want to customize the traffic-control profile configuration on Virtual Chassis ports. To do so, you can configure an output traffic-control profile and apply it to all Virtual Chassis ports interfaces in the Virtual Chassis.

This topic provides an overview of the default CoS configuration for Virtual Chassis ports and helps you understand the components of the CoS configuration that you can customize.

- Default CoS Configuration for Virtual Chassis Ports on page 145
- Supported Platforms and Maximums for CoS Configuration of Virtual Chassis Ports on page 146
- Default Classifiers for Virtual Chassis Ports on page 147
- Default Rewrite Rules for Virtual Chassis Ports on page 147
- Default Scheduler Map for Virtual Chassis Ports on page 148
- Customized CoS Configuration for Virtual Chassis Ports on page 149

Default CoS Configuration for Virtual Chassis Ports

In an MX Series Virtual Chassis configuration, the Virtual Chassis ports behave like switch fabric ports to transport packets between the member routers in a Virtual Chassis. More

specifically, the Virtual Chassis ports carry internal control traffic within the Virtual Chassis and forward user traffic between line cards in the router.

Like traffic on standard network port interfaces, traffic on Virtual Chassis port interfaces is mapped to one of four forwarding classes, as follows:

- Internal Virtual Chassis Control Protocol (VCCP) traffic is mapped to the network control forwarding class with the code point (IEEE 802.1p bit) value set to '111'b. You cannot change this configuration.
- Control traffic is mapped to the network control forwarding class with the code point (IEEE 802.1p bit) value set to '110'b. You cannot change this configuration.
- User traffic is mapped to the best effort, expedited forwarding, and assured forwarding traffic classes.

The CoS configuration applies globally to all Virtual Chassis ports in the Virtual Chassis. You cannot configure CoS for an individual Virtual Chassis port (such as **vcp-2/2/0**). If you create a new Virtual Chassis port, the global CoS configuration is propagated to the newly created Virtual Chassis port when the member router on which the new Virtual Chassis port resides joins the Virtual Chassis. Alternatively, you can configure CoS for the Virtual Chassis ports by configuring CoS for a standard network port, and then converting the network port to a Virtual Chassis port by issuing the **request virtual-chassis vc-port set** command.

You can convert a standard network port (for example, **xe-2/2/1**) to a Virtual Chassis port by issuing the **request virtual-chassis vc-port set** command. If the standard network port was configured with different CoS settings than the CoS configuration in effect for all Virtual Chassis ports in the Virtual Chassis, the newly converted Virtual Chassis port (**vcp-2/2/1**) uses the CoS configuration defined for all Virtual Chassis port interfaces instead of the original CoS configuration associated with the network port.

The default CoS configuration for Virtual Chassis ports provides the following benefits to keep the Virtual Chassis operating properly:

- Gives preference to internal VCCP traffic that traverses the Virtual Chassis port interfaces
- Prioritizes control traffic over user traffic on the Virtual Chassis port interfaces
- Preserves the CoS properties of each packet as it travels between member routers in the Virtual Chassis

Supported Platforms and Maximums for CoS Configuration of Virtual Chassis Ports

You can configure Virtual Chassis ports only on Modular Port Concentrator/Modular Interface Card (MPC/MIC) interfaces in the following MX Series 3D Universal Edge Routers with dual Routing Engines:

- MX240 3D Universal Edge Router
- MX480 3D Universal Edge Router
- MX960 3D Universal Edge Router

MPC/MIC interfaces support the following maximums for forwarding classes and priority scheduling levels:

- Up to eight forwarding classes
- Up to five priority scheduling levels

Default Classifiers for Virtual Chassis Ports

Classification takes place when a packet enters a Virtual Chassis member router from a network port. For Virtual Chassis configurations that support more than two member routers, the packet is reclassified for CoS treatment according to the default IEEE 802.1p classifier rules that apply to the Virtual Chassis port as the packet travels through the intermediate member routers in the Virtual Chassis. When the packet enters the last member router in the Virtual Chassis, it is reclassified according to the original classifier rules that applied when the packet entered the Virtual Chassis from a network port.

NOTE: This reclassification behavior does not apply to an MX Series Virtual Chassis, which supports only two member routers in the current release.

Because there are no intermediate member routers between the two member routers in an MX Series Virtual Chassis, the packet is not reclassified according to the default classifier rules for the Virtual Chassis port. Instead, the original classifier rules that applied when the packet entered the Virtual Chassis on a network port are retained.

The default IEEE 802.1p classifier rules map the code point (or .1p bit) value to the forwarding class and loss priority. You can display the default IEEE 802.1p classifier rules by issuing the **show class-of-service classifier** command:

```
{master:member0-re0}

user@host> show class-of-service classifier type ieee-802.1
Classifier: ieee8021p-default, Code point type: ieee-802.1, Index: 11
  Code point         Forwarding class                  Loss priority
  000                best-effort                       low
  001                best-effort                       high
  010                expedited-forwarding              low
  011                expedited-forwarding              high
  100                assured-forwarding                low
  101                assured-forwarding                high
  110                network-control                   low
  111                network-control                   high
```

Default Rewrite Rules for Virtual Chassis Ports

When a packet enters the Virtual Chassis from a network port, normal CoS classification takes place. If the packet exits a member router through the Virtual Chassis port to the other member router, the CoS software encapsulates the packet with a virtual LAN (VLAN) tag that contains the code point information used for CoS treatment. The code point value is assigned according to the default IEEE 802.1p rewrite rules, which map the forwarding class and loss priority value to a code point value.

You can display the default IEEE 802.1p rewrite rules by issuing the **show class-of-service rewrite-rule** command:

```
{master:member0-re0}

user@host> show class-of-service rewrite-rule type ieee-802.1
Rewrite rule: ieee8021p-default, Code point type: ieee-802.1, Index: 34
  Forwarding class                  Loss priority       Code point
  best-effort                       low                 000
  best-effort                       high                001
  expedited-forwarding              low                 010
  expedited-forwarding              high                011
  assured-forwarding                low                 100
  assured-forwarding                high                101
  network-control                   low                 110
  network-control                   high                111
```

Default Scheduler Map for Virtual Chassis Ports

When you create a Virtual Chassis port, it automatically functions as a hierarchical scheduler. However, you cannot explicitly configure hierarchical scheduling on Virtual Chassis ports.

Virtual Chassis ports use the same default scheduler used by standard network ports. The network control and best effort forwarding classes are both assigned low priority, and only 5 percent of the bandwidth is allocated to control traffic.

You can display the scheduler parameters and the mapping of schedulers to forwarding classes by issuing the **show class-of-service scheduler-map** command. For brevity, the following example shows only the portions of the output relevant to the default best effort (**default-be**) and default network control (**default-nc**) schedulers.

```
{master:member0-re0}

user@host> show class-of-service scheduler-map
Scheduler map: <default>, Index: 2

  Scheduler: <default-be>, Forwarding class: best-effort, Index: 21
    Transmit rate: 95 percent, Rate Limit: none, Buffer size: 95 percent, Buffer
 Limit: none, Priority: low
    Excess Priority: low
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             any             1    <default-drop-profile>
      Medium low      any             1    <default-drop-profile>
      Medium high     any             1    <default-drop-profile>
      High            any             1    <default-drop-profile>

  Scheduler: <default-nc>, Forwarding class: network-control, Index: 23
    Transmit rate: 5 percent, Rate Limit: none, Buffer size: 5 percent, Buffer
Limit: none, Priority: low
    Excess Priority: low
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             any             1    <default-drop-profile>
      Medium low      any             1    <default-drop-profile>
      Medium high     any             1    <default-drop-profile>
```

```
    High            any             1     <default-drop-profile>
...
```

Customized CoS Configuration for Virtual Chassis Ports

Depending on your network topology, you might want to customize the CoS configuration for Virtual Chassis ports. For example, you might want to allocate more than the default 5 percent of the Virtual Chassis port bandwidth to control traffic. Or, you might want to assign different priorities and excess rates to different forwarding classes.

Output Traffic-Control Profiles

To create a customized (nondefault) CoS configuration and apply it to all Virtual Chassis ports, you can configure an output traffic-control profile, which defines a set of traffic scheduling resources and references a scheduler map. You then apply the profile to all Virtual Chassis port interfaces. To apply the output traffic-control profile globally to all Virtual Chassis port interfaces, you must use **vcp-*** as the interface name representing all Virtual Chassis port interfaces. You cannot configure CoS for an individual Virtual Chassis port (such as **vcp-1/1/0**).

For an example that shows how to configure an output traffic-control profile customized for Virtual Chassis ports, see "Example: Configuring Class of Service for Virtual Chassis Ports on MX Series 3D Universal Edge Routers" on page 150.

Classifiers and Rewrite Rules

Configuring nondefault IEEE 802.1p ingress classifiers and IEEE 802.1p egress rewrite rules *has no effect* in a two-member MX Series Virtual Chassis.

Because there are no intermediate routers between the two member routers in an MX Series Virtual Chassis, packets are not reclassified according to the default classifier rules for Virtual Chassis ports. Instead, the original classifier rules that applied when the packet entered the Virtual Chassis on a network port are retained, making configuration of nondefault ingress classifiers and nondefault egress rewrite rules unnecessary in the current release.

Per-Priority Shaping

MPC/MIC interfaces support per-priority shaping, which enables you to configure a separate traffic shaping rate for each of the five priority scheduling levels. However, configuring per-priority shaping for Virtual Chassis ports on MPC/MIC interfaces is unnecessary for the following reasons:

- The neighboring member router has exactly the same bandwidth.
- The same type of Virtual Chassis port is present at both ends of the connection.

Related Documentation

- Guidelines for Configuring Class of Service for Virtual Chassis Ports on page 150
- Example: Configuring Class of Service for Virtual Chassis Ports on MX Series 3D Universal Edge Routers on page 150
- *Junos OS Class of Service Library for Routing Devices*

Guidelines for Configuring Class of Service for Virtual Chassis Ports

Consider the following guidelines when you configure class of service (CoS) for Virtual Chassis ports in an MX Series Virtual Chassis:

- Virtual Chassis ports on MPC/MIC interfaces support a maximum of eight forwarding classes and five priority scheduling levels.
- The same CoS configuration applies globally to all Virtual Chassis ports in the Virtual Chassis. You cannot configure CoS for an individual Virtual Chassis port (such as **vcp-3/1/0**).
- The CoS configuration is propagated to a newly created Virtual Chassis port as soon as the member router on which the new Virtual Chassis port resides joins the Virtual Chassis.
- Although Virtual Chassis ports function as hierarchical schedulers, you cannot explicitly configure hierarchical scheduling on Virtual Chassis ports.
- If you configure a nondefault output traffic-control profile to customize the CoS configuration, you must apply the profile to all Virtual Chassis port interfaces at once by using **vcp-*** as the interface name.
- Configuring nondefault IEEE 802.1p ingress classifiers and IEEE 802.1p egress rewrite rules has no effect in a two-member MX Series Virtual Chassis because the forwarding class assigned to a packet is maintained across the Virtual Chassis until the packet reaches the egress network port.
- Configuring per-priority shaping for Virtual Chassis ports is unnecessary because the neighboring member router has exactly the same bandwidth, and the same type of Virtual Chassis port is present at both ends of the connection.

Related Documentation

- Class of Service Overview for Virtual Chassis Ports on page 145
- Example: Configuring Class of Service for Virtual Chassis Ports on MX Series 3D Universal Edge Routers on page 150
- *Junos OS Class of Service Library for Routing Devices*

Example: Configuring Class of Service for Virtual Chassis Ports on MX Series 3D Universal Edge Routers

This example illustrates a typical class of service (CoS) configuration that you might want to use for the Virtual Chassis ports in an MX Series Virtual Chassis.

- Requirements on page 151
- Overview on page 151
- Configuration on page 152

Requirements

Before you begin:

- Configure a Virtual Chassis consisting of two MX Series routers.

  See "Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis" on page 75

Overview

By default, all Virtual Chassis ports in an MX Series Virtual Chassis use a default CoS configuration specifically tailored for Virtual Chassis ports. The default configuration, which applies to all Virtual Chassis ports in the Virtual Chassis, includes classifiers, forwarding classes, rewrite rules, and schedulers. This default CoS configuration prioritizes internal Virtual Chassis Control Protocol (VCCP) traffic that traverses the Virtual Chassis port interfaces, and prioritizes control traffic over user traffic on the Virtual Chassis ports. In most cases, the default CoS configuration is adequate for your needs without requiring any additional CoS configuration.

In some cases, however, you might want to customize the traffic-control profile configuration on Virtual Chassis ports. For example, you might want to assign different priorities and excess rates to different forwarding classes. To create a nondefault CoS configuration, you can create an output traffic-control profile that defines a set of traffic scheduling resources and references a scheduler map. You then apply the output traffic-control profile to all Virtual Chassis port interfaces at once by using **vcp-*** as the interface name representing all Virtual Chassis ports. You cannot configure CoS for Virtual Chassis ports on an individual basis.

Table 19 on page 151 shows the nondefault CoS scheduler hierarchy configured in this example for the Virtual Chassis ports.

Table 19: Sample CoS Scheduler Hierarchy for Virtual Chassis Ports

Traffic Type	Queue Number	Priority	Transmit Rate/ Excess Rate
Network control (VCCP traffic)	3	Medium	90%
Expedited forwarding (voice traffic)	2	High	10%
Assured forwarding (video traffic)	1	Excess Low	99%
Best effort (data traffic)	0	Excess Low	1%

In this example, you create a nondefault CoS configuration for Virtual Chassis ports by completing the following tasks on the Virtual Chassis master router:

- Associate forwarding classes with **queue 0** through **queue 3**, and configure a fabric priority value for each queue.
- Configure an output traffic control profile named **tcp-vcp-ifd** to define traffic scheduling parameters, and associate a scheduler map named **sm-vcp-ifd** with the traffic control profile.
- Apply the output traffic-control profile to the **vcp-*** interface, which represents all Virtual Chassis port interfaces in the Virtual Chassis.
- Associate the **sm-vcp-ifd** scheduler map with the forwarding classes and scheduler configuration.
- Configure the parameters for schedulers **s-medium-priority**, **s-high-priority**, **s-low-priority**, **s-high-weight**, and **s-low-weight**.

Configuration

CLI Quick Configuration

To quickly create a nondefault CoS configuration for Virtual Chassis ports, copy the following commands and paste them into the router terminal window:

```
[edit]
set class-of-service forwarding-classes queue 0 best-effort
set class-of-service forwarding-classes queue 0 priority low
set class-of-service forwarding-classes queue 1 assured-forwarding
set class-of-service forwarding-classes queue 1 priority low
set class-of-service forwarding-classes queue 2 expedited-forwarding
set class-of-service forwarding-classes queue 2 priority high
set class-of-service forwarding-classes queue 3 network-control
set class-of-service forwarding-classes queue 3 priority high
set class-of-service traffic-control-profiles tcp-vcp-ifd scheduler-map sm-vcp-ifd
set class-of-service interfaces vcp-* output-traffic-control-profile tcp-vcp-ifd
set class-of-service scheduler-maps sm-vcp-ifd forwarding-class network-control
  scheduler s-medium-priority
set class-of-service scheduler-maps sm-vcp-ifd forwarding-class expedited-forwarding
  scheduler s-high-priority
set class-of-service scheduler-maps sm-vcp-ifd forwarding-class assured-forwarding
  scheduler s-high-weight
set class-of-service scheduler-maps sm-vcp-ifd forwarding-class best-effort scheduler
  s-low-weight
set class-of-service schedulers s-medium-priority transmit-rate percent 90
set class-of-service schedulers s-medium-priority priority medium-high
set class-of-service schedulers s-medium-priority excess-priority high
set class-of-service schedulers s-high-priority transmit-rate percent 10
set class-of-service schedulers s-high-priority priority high
set class-of-service schedulers s-high-priority excess-priority high
set class-of-service schedulers s-low-priority priority low
set class-of-service schedulers s-high-weight excess-rate percent 99
set class-of-service schedulers s-low-weight excess-rate percent 1
```

Step-by-Step Procedure

To create a nondefault CoS configuration for Virtual Chassis ports in an MX Series Virtual Chassis:

1. Log in to the console on the master router of the Virtual Chassis.

2. Specify that you want to configure CoS forwarding classes.

```
{master:member0-re0} [edit]
user@host# edit class-of-service forwarding-classes
```

3. Associate a forwarding class with each queue name and number, and configure a fabric priority value for each queue.

```
{master:member0-re0} [edit class-of-service forwarding-classes]
user@host# set queue 0 best-effort priority low
user@host# set queue 1 assured-forwarding priority low
user@host# set queue 2 expedited-forwarding priority high
user@host# set queue 3 network-control priority high
```

4. Return to the **[edit class-of-service]** hierarchy level to configure an output traffic-control profile.

```
{master:member0-re0} [edit class-of-service forwarding-classes]
user@host# up
```

5. Configure an output traffic-control profile and associate it with a scheduler map.

```
{master:member0-re0} [edit class-of-service]
user@host# set traffic-control-profiles tcp-vcp-ifd scheduler-map sm-vcp-ifd
```

6. Apply the output traffic-control profile to all Virtual Chassis port interfaces in the Virtual Chassis.

```
{master:member0-re0} [edit class-of-service]
user@host# set interfaces vcp-* output-traffic-control-profile tcp-vcp-ifd
```

7. Specify that you want to configure the scheduler map.

```
{master:member0-re0} [edit class-of-service]
user@host# edit scheduler-maps sm-vcp-ifd
```

8. Associate the scheduler map with the scheduler configuration and forwarding classes.

```
{master:member0-re0} [edit class-of-service scheduler-maps sm-vcp-ifd]
user@host# set forwarding-class network-control scheduler s-medium-priority
user@host# set forwarding-class expedited-forwarding scheduler s-high-priority
user@host# set forwarding-class assured-forwarding scheduler s-high-weight
user@host# set forwarding-class best-effort scheduler s-low-weight
```

9. Return to the **[edit class-of-service]** hierarchy level to configure the schedulers.

```
{master:member0-re0} [edit class-of-service scheduler-maps sm-vcp-ifd]
```

```
user@host# up 2
```

10. Configure parameters for the schedulers.

```
{master:member0-re0} [edit class-of-service]
user@host# set schedulers s-medium-priority priority medium-high excess-priority
  high transmit-rate percent 90
user@host# set schedulers s-high-priority priority high excess-priority high
  transmit-rate percent 10
user@host# set schedulers s-low-priority priority low
user@host# set schedulers s-high-weight excess-rate percent 99
user@host# set schedulers s-low-weight excess-rate percent 1
```

Results From the **[edit class-of-service]** hierarchy level in configuration mode, confirm the results of your configuration by issuing the **show** statement. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
{master:member0-re0} [edit class-of-service]
user@host# show
forwarding-classes {
  queue 0 best-effort priority low;
  queue 1 assured-forwarding priority low;
  queue 2 expedited-forwarding priority high;
  queue 3 network-control priority high;
}
traffic-control-profiles {
  tcp-vcp-ifd {
    scheduler-map sm-vcp-ifd;
  }
}
interfaces {
  vcp-* {
    output-traffic-control-profile tcp-vcp-ifd;
  }
}
scheduler-maps {
  sm-vcp-ifd {
    forwarding-class network-control scheduler s-medium-priority;
    forwarding-class expedited-forwarding scheduler s-high-priority;
    forwarding-class assured-forwarding scheduler s-high-weight;
    forwarding-class best-effort scheduler s-low-weight;
  }
}
schedulers {
  s-medium-priority {
    transmit-rate percent 90;
    priority medium-high;
    excess-priority high;
  }
  s-high-priority {
    transmit-rate percent 10;
    priority high;
```



```
        excess-priority high;
    }
    s-low-priority {
        priority low;
    }
    s-high-weight {
        excess-rate percent 99;
    }
    s-low-weight {
        excess-rate percent 1;
    }
}
```

If you are done configuring CoS on the master router, enter **commit** from configuration mode.

Related Documentation

- Class of Service Overview for Virtual Chassis Ports on page 145
- Guidelines for Configuring Class of Service for Virtual Chassis Ports on page 150
- *Junos OS Class of Service Library for Routing Devices*

CHAPTER 7

Configuring Redundancy Mechanisms on Aggregated Ethernet Interfaces in a Virtual Chassis

- Redundancy Mechanisms on Aggregated Ethernet Interfaces in a Virtual Chassis on page 157
- Configuring Module Redundancy for a Virtual Chassis on page 159
- Configuring Chassis Redundancy for a Virtual Chassis on page 161
- Multichassis Link Aggregation in a Virtual Chassis on page 162
- Targeted Traffic Distribution on Aggregated Ethernet Interfaces in a Virtual Chassis on page 163

Redundancy Mechanisms on Aggregated Ethernet Interfaces in a Virtual Chassis

Starting in Junos OS Release 13.2, an MX Series Virtual Chassis configured with targeted traffic distribution for IP demux or VLAN demux subscribers on aggregated Ethernet interfaces supports three types of redundancy mechanisms: link redundancy, module redundancy, and chassis redundancy.

- Link Redundancy in a Virtual Chassis on page 157
- Module Redundancy in a Virtual Chassis on page 158
- Chassis Redundancy in a Virtual Chassis on page 158

Link Redundancy in a Virtual Chassis

By default, the router uses *link redundancy*, also known as *port redundancy*, as the default redundancy mechanism for targeted distribution on aggregated Ethernet interfaces. With link redundancy, the router assigns backup links for a subscriber based on the link with the fewest number of subscribers.

In an MX Series Virtual Chassis configured with link redundancy, the primary link and backup link can be assigned on the same Modular Port Concentrator/Modular Interface Card (MPC/MIC) module, on different MPC/MIC modules in the same member router, or on different MPC/MIC modules in different member routers. This feature provides redundancy if a link in the MX Series Virtual Chassis configuration fails.

Because link redundancy is the default redundancy mechanism, no special configuration is required on the Virtual Chassis master router to enable it.

Module Redundancy in a Virtual Chassis

You can configure *module redundancy*, also known as *Flexible PIC Concentrator (FPC) redundancy*, to provide redundancy if a module or a link fails. The router assigns backup links for the subscriber interface on a different MPC/MIC module from the primary link, based on the link with the fewest number of subscribers among the links on different modules.

In an MX Series Virtual Chassis configured with link redundancy, the router assigns the primary link and backup link to different MPC/MIC modules. For purposes of link selection, the router gives all MPC/MIC modules in the Virtual Chassis equal weight, and disregards the role (master or backup) of the member router in which the MPC/MIC module is installed. The router uses an algorithm to assign the primary and backup links, and is as likely to assign a primary link to an MPC/MIC module in the Virtual Chassis master router as it is to assign the primary link to an MPC/MIC module in the Virtual Chassis backup router.

Chassis Redundancy in a Virtual Chassis

Unlike link redundancy and module redundancy, which are supported on both standalone routers and Virtual Chassis member routers, chassis redundancy is available only for member routers in an MX Series Virtual Chassis configuration.

Chassis redundancy and module redundancy use the same algorithm for link assignment, with the exception that in a Virtual Chassis with chassis redundancy configured, the router assigns the backup link to an MPC/MIC module in a member router *other* than the router on which the primary link resides. For example, in a two-member MX Series Virtual Chassis, if the primary link for the aggregated Ethernet bundle is assigned to an MPC/MIC module in the Virtual Chassis master router, the router assigns the backup link to an MPC/MIC module in the Virtual Chassis backup router.

Chassis redundancy provides protection if the MPC/MIC module containing the primary link fails. In this event, the subscriber connections fail over to the backup link on the MPC/MIC module in the other member router.

BEST PRACTICE: **We recommend that you do not configure both module (FPC) redundancy and chassis redundancy for the same aggregated Ethernet interface in an MX Series Virtual Chassis. If you do, module redundancy takes precedence over chassis redundancy.**

Release History Table

Release	Description
13.2	Starting in Junos OS Release 13.2, an MX Series Virtual Chassis configured with targeted traffic distribution for IP demux or VLAN demux subscribers on aggregated Ethernet interfaces supports three types of redundancy mechanisms: link redundancy, module redundancy, and chassis redundancy.

Related Documentation

- Targeted Traffic Distribution on Aggregated Ethernet Interfaces in a Virtual Chassis on page 163
- Configuring Module Redundancy for a Virtual Chassis on page 159
- Configuring Chassis Redundancy for a Virtual Chassis on page 161

Configuring Module Redundancy for a Virtual Chassis

By default, a router or switch uses link redundancy for aggregated Ethernet interfaces (bundles) configured with targeted traffic distribution. Starting in Junos OS Release 13.2, as an alternative to using link redundancy, you can configure module redundancy, also known as FPC redundancy, for a Virtual Chassis configured with targeted traffic distribution for IP demux or VLAN demux subscribers on aggregated Ethernet interfaces.

In a Virtual Chassis, module redundancy assigns the primary link and backup link to *different* MPC/MIC modules or line cards, regardless of the Virtual Chassis role (master or backup) of the member device in which the module is installed. Module redundancy provides redundancy protection if a module or a link in the Virtual Chassis fails.

Before you begin:

- Configure a Virtual Chassis consisting of two MX Series routers or two EX9200 switches.

  See "Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis" on page 75

- Ensure that the aggregated Ethernet bundle is configured *without* link protection.

  See *Configuring Aggregated Ethernet Link Protection*

To configure module redundancy:

1. Log in to the console on the master device of the Virtual Chassis.

2. Specify that you want to configure the demux logical interface.

   ```
   {master:member0-re0} [edit]
   user@host# edit interfaces demux0 unit logical-unit-number
   ```

3. Enable targeted distribution for the interface.

   ```
   {master:member0-re0} [edit intefaces demux0 unit logical-unit-number]
   user@host# set targeted-distribution
   ```

4. Specify the aggregated Ethernet bundle for which you want to configure module redundancy.

   {master:member0-re0} [edit]
   user@host# **edit interfaces ae*number*** aggregated-ether-options

5. Enable module (FPC) redundancy for the specified aggregated Ethernet bundle.

   {master:member0-re0} [edit interfaces ae*number* aggregated-ether-options]
   user@host# **set** logical-interface-fpc-redundancy

BEST PRACTICE: We recommend that you do not configure both module (FPC) redundancy and chassis redundancy for the same aggregated Ethernet interface in the Virtual Chassis. If you do, module redundancy takes precedence over chassis redundancy.

Release History Table

Release	Description
13.2	Starting in Junos OS Release 13.2, as an alternative to using link redundancy, you can configure module redundancy, also known as FPC redundancy, for a Virtual Chassis configured with targeted traffic distribution for IP demux or VLAN demux subscribers on aggregated Ethernet interfaces.

Related Documentation

- Targeted Traffic Distribution on Aggregated Ethernet Interfaces in a Virtual Chassis on page 163
- Redundancy Mechanisms on Aggregated Ethernet Interfaces in a Virtual Chassis on page 157
- Configuring Chassis Redundancy for a Virtual Chassis on page 161

Configuring Chassis Redundancy for a Virtual Chassis

By default, the router uses link redundancy for aggregated Ethernet interfaces (bundles) configured with targeted traffic distribution. Starting in Junos OS Release 13.2, as an alternative to using link redundancy, you can configure chassis redundancy for an MX Series Virtual Chassis configured with targeted traffic distribution for IP demux or VLAN demux subscribers on aggregated Ethernet interfaces.

In an MX Series Virtual Chassis, chassis redundancy assigns the backup link to an MPC/MIC module in a member router *other* than the member router on which the primary link resides. For example, in a two-member MX Series Virtual Chassis where the primary link for the aggregated Ethernet bundle is on an MPC/MIC module in the master router, chassis redundancy assigns the backup link for the bundle to an MPC/MIC module in the backup router. Chassis redundancy provides protection if the MPC/MIC module containing the primary link fails. In this event, the subscriber connections fail over to the backup link on the MPC/MIC module in the other member router.

Unlike link redundancy and module redundancy, each of which are supported for both standalone routers and Virtual Chassis member routers, chassis redundancy is available only for member routers in an MX Series Virtual Chassis.

Before you begin:

- Configure a Virtual Chassis consisting of two MX Series routers.

  See "Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis" on page 75

- Ensure that the aggregated Ethernet bundle is configured *without* link protection.

  See *Configuring Aggregated Ethernet Link Protection*

To configure chassis redundancy for an MX Series Virtual Chassis:

1. Log in to the console on the master router of the Virtual Chassis.

2. Specify that you want to configure the demux logical interface.

   ```
   {master:member0-re0} [edit]
   user@host# edit interfaces demux0 unit logical-unit-number
   ```

3. Enable targeted distribution for the interface.

   ```
   {master:member0-re0} [edit intefaces demux0 unit logical-unit-number]
   user@host# set targeted-distribution
   ```

4. Specify the aggregated Ethernet bundle for which you want to configure chassis redundancy.

   ```
   {master:member0-re0} [edit]
   user@host# edit interfaces aenumber aggregated-ether-options
   ```

5. Enable module (FPC) redundancy for the specified aggregated Ethernet bundle.

   {master:member0-re0} [edit interfaces ae*number* aggregated-ether-options]
   user@host# **set** logical-interface-chassis-redundancy

BEST PRACTICE: We recommend that you do not configure both module (FPC) redundancy and chassis redundancy for the same aggregated Ethernet interface in an MX Series Virtual Chassis. If you do, module redundancy takes precedence over chassis redundancy.

Release History Table

Release	Description
13.2	Starting in Junos OS Release 13.2, as an alternative to using link redundancy, you can configure chassis redundancy for an MX Series Virtual Chassis configured with targeted traffic distribution for IP demux or VLAN demux subscribers on aggregated Ethernet interfaces.

Related Documentation

- Targeted Traffic Distribution on Aggregated Ethernet Interfaces in a Virtual Chassis on page 163
- Redundancy Mechanisms on Aggregated Ethernet Interfaces in a Virtual Chassis on page 157
- Configuring Module Redundancy for a Virtual Chassis on page 159

Multichassis Link Aggregation in a Virtual Chassis

Starting in Junos OS Release 13.3, you can configure multichassis link aggregation (MC-LAG) in an MX Series Virtual Chassis.

MC-LAG enables a device to form a logical link aggregation group interface with two or more other devices. The MC-LAG devices use the Inter-Chassis Communication Protocol (ICCP) to exchange control information between two MC-LAG network devices.

When you configure MC-LAG with an MX Series Virtual Chassis, the link aggregation group spans links to two Virtual Chassis configurations. Each Virtual Chassis consists of two MX Series member routers that form a logical system managed as a single network element. ICCP exchanges control information between the global master router (VC-M) of the first Virtual Chassis and the VC-M of the second Virtual Chassis.

To configure MC-LAG on member routers in a Virtual Chassis, use the same procedure that you would use to configure MC-LAG on a standalone MX Series router.

NOTE: Internet Group Management Protocol (IGMP) snooping is not supported on MC-LAG interfaces in an MX Series Virtual Chassis.

Release History Table

Release	Description
13.3	Starting in Junos OS Release 13.3, you can configure multichassis link aggregation (MC-LAG) in an MX Series Virtual Chassis.

Related Documentation

- *Configuring Multichassis Link Aggregation on MX Series Routers*
- Global Roles and Local Roles in a Virtual Chassis on page 30

Targeted Traffic Distribution on Aggregated Ethernet Interfaces in a Virtual Chassis

By default, member routers or switches in an EX9200 or MX Series Virtual Chassis use hash-based traffic distribution for subscriber interfaces in aggregated Ethernet bundles configured without link protection. The hash-based model distributes subscriber interface traffic over multiple links in the bundle, enabling you to load balance multiple traffic flows through the logical subscriber interface.

Starting in Junos OS Release 13.2, as an alternative to using hash-based distribution in an EX9200 or MX Series Virtual Chassis, you can configure targeted traffic distribution for IP demultiplexing (demux) or VLAN demux subscriber interfaces in an aggregated Ethernet bundle that is configured without link protection.

- Targeted Distribution in a Virtual Chassis on page 163
- Benefits of Targeted Distribution on page 163

Targeted Distribution in a Virtual Chassis

Targeted distribution enables you to configure the Virtual Chassis to send (target) all egress data traffic for a logical subscriber interface across a single member link in an *aggregated Ethernet bundle*, also referred to as an IEEE 802.3ad link aggregation group (LAG) bundle. You configure targeted distribution for a demux subscriber interface on the Virtual Chassis master router or switch.

With targeted distribution, the router or switch in a Virtual Chassis assigns the primary member link and backup member link for the aggregated Ethernet bundle across *all* Virtual Chassis port links that belong to the aggregated Ethernet bundle. To accomplish load balancing, the router or switch evenly distributes the demux subscriber interfaces over these member links.

Benefits of Targeted Distribution

Targeted distribution is especially useful in a Virtual Chassis configuration in which subscriber traffic enters through a Virtual Chassis port on one member router or switch and exits through a Virtual Chassis port on a different member router or switch. By combining Virtual Chassis ports from different member router or switches as member links of the aggregated Ethernet bundle, targeted distribution provides increased redundancy in the event of a chassis or link failure.

Release History Table

Release	Description
13.2	Starting in Junos OS Release 13.2, as an alternative to using hash-based distribution in an EX9200 or MX Series Virtual Chassis, you can configure targeted traffic distribution for IP demultiplexing (demux) or VLAN demux subscriber interfaces in an aggregated Ethernet bundle that is configured without link protection.

Related Documentation

- Redundancy Mechanisms on Aggregated Ethernet Interfaces in a Virtual Chassis on page 157
- Configuring Module Redundancy for a Virtual Chassis on page 159
- Configuring Chassis Redundancy for a Virtual Chassis on page 161

CHAPTER 8

Upgrading Junos OS in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers by Rebooting the Routing Engines

- Example: Upgrading Junos OS in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers by Rebooting the Routing Engines on page 165

Example: Upgrading Junos OS in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers by Rebooting the Routing Engines

You can upgrade an MX Series Virtual Chassis configuration from Junos OS Release 11.2 to a later release by rebooting each of the Routing Engines. Both member routers in the Virtual Chassis must have dual Routing Engines installed.

NOTE: **Make sure all four Routing Engines in the Virtual Chassis (both Routing Engines in the master router and both Routing Engines in the backup router) are running the same Junos OS release.**

This example describes how to upgrade Junos OS in a two-member MX Series Virtual Chassis by rebooting the Routing Engines. For information about upgrading Junos OS in an MX Series Virtual Chassis by performing a unified ISSU, see "Upgrading Junos OS in an MX Series Virtual Chassis by Performing a Unified ISSU" on page 177.

- Requirements on page 165
- Overview and Topology on page 166
- Configuration on page 167

Requirements

This example uses the following software and hardware and components:

- Junos OS Release 12.3 and later releases
- One MX240 3D Universal Edge Router with dual Routing Engines

- One MX480 3D Universal Edge Router with dual Routing Engines

NOTE: This configuration example has been tested using the software release listed and is assumed to work on all later releases.

See Table 20 on page 167 for information about the hardware installed in each MX Series router.

BEST PRACTICE: We recommend that you use the **commit synchronize** command to save any configuration changes to the Virtual Chassis.

For an MX Series Virtual Chassis, the **force** option is the default and only behavior when you issue the **commit synchronize** command. Issuing the **commit synchronize** command for an MX Series Virtual Chassis configuration has the same effect as issuing the **commit synchronize force** command.

Overview and Topology

To upgrade Junos OS in an MX Series Virtual Chassis configuration by rebooting the Routing Engines, you must:

1. Prepare for the upgrade.
2. Install the Junos OS software package on each of the four Routing Engines.
3. Re-enable graceful Routing Engine switchover and nonstop active routing.
4. Reboot the Routing Engines to run the new Junos OS release.

This example upgrades Junos OS in an MX Series Virtual Chassis configuration that uses the basic topology shown in Figure 5 on page 166. For redundancy, each member router is configured with two Virtual Chassis ports.

Figure 5: Sample Topology for a Virtual Chassis with Two MX Series Routers

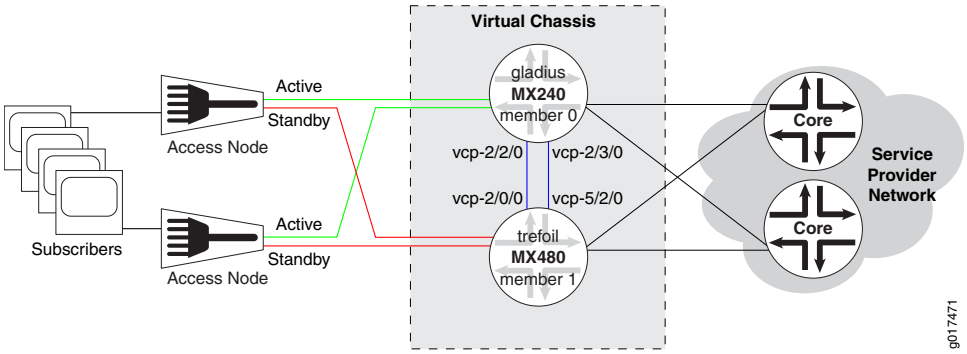


Table 20 on page 167 shows the hardware and software configuration settings for each MX Series router in the Virtual Chassis.

Table 20: Components of the Sample MX Series Virtual Chassis

Router Name	Hardware	Serial Number	Member ID	Role	Virtual Chassis Ports	Network Port Slot Numbering
gladius	MX240 router with: • 60-Gigabit Ethernet Enhanced Queuing MPC • 20-port Gigabit Ethernet MIC with SFP • 4-port 10-Gigabit Ethernet MIC with XFP • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member0-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member0-re1**)	JN10C7135AFC	0	routing-engine (master)	vcp-2/2/0 vcp-2/3/0	FPC 0 – 11
trefoil	MX480 router with: • Two 30-Gigabit Ethernet Queuing MPCs • Two 20-port Gigabit Ethernet MICs with SFP • Two 2-port 10-Gigabit Ethernet MICs with XFP • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member1-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member1-re1**)	JN115D117AFB	1	routing-engine (backup)	vcp-2/0/0 vcp-5/2/0	FPC 12 – 23 (offset = 12)

Configuration

To upgrade Junos OS in a two-member MX Series Virtual Chassis by rebooting the Routing Engines, perform these tasks:

- Preparing for the Upgrade on page 168
- Installing the Junos OS Software Package on Each Routing Engine on page 169

- Re-enabling Graceful Routing Engine Switchover and Nonstop Active Routing on page 170
- Rebooting the Routing Engines to Run the New Junos OS Release on page 170

Preparing for the Upgrade

Step-by-Step Procedure

To prepare for the upgrade process:

1. Use FTP or a Web browser to download the Junos OS software package to the master Routing Engine on the Virtual Chassis master router (VC-M).

   See *Downloading Software*.

2. Disable nonstop active routing on the master router.

   ```
   {master:member0-re0}[edit routing-options]
   user@gladius# deactivate nonstop-routing
   ```

3. Disable graceful Routing Engine switchover on the master router.

   ```
   {master:member0-re0}[edit chassis redundancy]
   user@gladius# deactivate graceful-switchover
   ```

4. Commit the configuration on the master router.

5. Exit CLI configuration mode.

   ```
   {master:member0-re0}[edit]
   user@gladius# exit
   ```

Installing the Junos OS Software Package on Each Routing Engine

Step-by-Step Procedure

Installing the Junos OS software package on each Routing Engine in an MX Series Virtual Chassis prepares the Routing Engines to run the new software release after a reboot. This action is also referred to as *arming* the Routing Engines.

To install the Junos OS software package on all four Routing Engines from the master router (**member0-re0**) in the Virtual Chassis:

- Install the software package on VC-Mm.

  ```
  {master:member0-re0}
  user@gladius> request system software add package-name
  ```

  On a properly formed Virtual Chassis, this command propagates the image to all four Routing Engines.

Results

Display the results of the installation. Verify that the correct software package has been installed on the local master Routing Engine in **member 0** (**member0-re0**) and on the local master Routing Engine in **member 1** (**member1-re0**).

```
user@gladius> show version invoke-on all-routing-engines
member0-re0:
--------------------------------------------------------------------------
Hostname: gladius
Model: mx240
. . .
JUNOS Installation Software [14.1R1.10]

member0-re1:
--------------------------------------------------------------------------
Hostname:gladius1
Model: mx240
. . .
JUNOS Installation Software [14.1R1.10]

member1-re0:
--------------------------------------------------------------------------
Hostname: trefoil
Model: mx240
. . .
JUNOS Installation Software [14.1R1.10]

member1-re1:
--------------------------------------------------------------------------
Hostname: trefoil1
Model: mx240
. . .
JUNOS Installation Software [14.1R1.10]
```

Re-enabling Graceful Routing Engine Switchover and Nonstop Active Routing

Step-by-Step Procedure

After upgrading the Junos OS release, you need to re-enable graceful Routing Engine switchover and nonstop active routing for the Virtual Chassis.

To re-enable graceful Routing Engine switchover and nonstop active routing from the Virtual Chassis master router (**member0-re0**):

1. In the console window on **member 0** (gladius), enable graceful Routing Engine switchover on the master router.

   ```
   {master:member0-re0}[edit chassis redundancy]
   user@gladius# activate graceful-switchover
   ```

2. Re-enable nonstop active routing on the master router.

   ```
   {master:member0-re0}[edit routing-options]
   user@gladius# activate nonstop-routing
   ```

3. Commit the configuration on the master router.

Rebooting the Routing Engines to Run the New Junos OS Release

Step-by-Step Procedure

NOTE: Rebooting both Routing Engines in the VC-M chassis may not result in a graceful switchover to the VC-B chassis, and is not recommended.

To reboot each of the four Routing Engines in an MX Series Virtual Chassis from the Virtual Chassis master router (**member0-re0**):

- Use the **request system reboot** command with no options.

  ```
  {master:member0-re0}
  user@gladius> request system reboot
  ```

  This command reboots all line cards in **member 0** (gladius) and **member 1** (trefoil) to use the new Junos OS release. A traffic disruption occurs until all line cards are back online and the Virtual Chassis re-forms.

Related Documentation

- Upgrading Junos OS in an MX Series Virtual Chassis by Performing a Unified ISSU on page 177
- Interchassis Redundancy and Virtual Chassis Overview on page 19
- Virtual Chassis Components Overview on page 24
- Switchover Behavior in an MX Series Virtual Chassis on page 34

CHAPTER 9

Upgrading Junos OS in an MX Series Virtual Chassis by Performing a Unified In-Service Software Upgrade (ISSU)

- Unified ISSU in a Virtual Chassis on page 171
- Preparing for a Unified ISSU in an MX Series Virtual Chassis on page 175
- Upgrading Junos OS in an MX Series Virtual Chassis by Performing a Unified ISSU on page 177

Unified ISSU in a Virtual Chassis

Starting in Junos OS Release 14.1, you can perform a unified in-service software upgrade (unified ISSU) for an MX Series Virtual Chassis configuration. Unified ISSU enables you to upgrade the Junos OS system software on the Virtual Chassis member routers with minimal traffic disruption and no disruption on the control plane.

This topic assumes that you are familiar with the global roles and local roles in an MX Series Virtual Chassis. For information, see "Global Roles and Local Roles in a Virtual Chassis" on page 30.

- Benefits of Performing a Unified ISSU in an Virtual Chassis on page 171
- Prerequisites for Performing a Unified ISSU in a Virtual Chassis on page 172
- How Unified ISSU Works in a Virtual Chassis on page 172
- Virtual Chassis Role Transitions After a Unified ISSU on page 173

Benefits of Performing a Unified ISSU in an Virtual Chassis

Performing a unified ISSU in an MX Series Virtual Chassis provides the following benefits:

- Upgrades the Junos OS software package while maintaining subscriber sessions.
- Reduces risk associated with a software upgrade. After performing a unified ISSU, the resulting system is exactly the same as if you had upgraded it with a system reboot.
- Prevents software upgrades from negatively affecting the service provider's ability to fulfill strict service-level agreements (SLAs).
- Eliminates network downtime during software image upgrades.

- Enables faster implementation of new Junos OS features.
- Provides feature parity with unified ISSU support on standalone MX Series routers.

Prerequisites for Performing a Unified ISSU in a Virtual Chassis

Before you start a unified ISSU in a two-member MX Series Virtual Chassis, make sure you do all of the following:

- Ensure that all four Routing Engines in the Virtual Chassis (both Routing Engines in the master router and both Routing Engines in the backup router) are running the same Junos OS software release.
- Back up the existing router configuration so you can revert (roll back) to it if necessary.
- Verify that both graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) are enabled.

How Unified ISSU Works in a Virtual Chassis

To perform a unified ISSU in an MX Series Virtual Chassis, you issue the **request system software in-service-upgrade *package-name*** command from the console window for the master Routing Engine in the Virtual Chassis master router (VC-Mm). Issuing this command from the VC-Mm copies the software package to all other Routing Engines in the Virtual Chassis.

The **request system software in-service-upgrade *package-name*** command functions the same for upgrading member routers in a Virtual Chassis configuration as it does for upgrading a standalone MX Series router with dual Routing Engines, *with the following exceptions*:

- The **no-copy**, **no-old-master-upgrade**, and **unlink** options for the **request system software in-service-upgrade** command are not available for an MX Series Virtual Chassis.
- The **reboot** option for the **request system software in-service-upgrade** command is accepted but ignored for an MX Series Virtual Chassis. A unified ISSU always reboots all Routing Engines in the Virtual Chassis member routers.

At a high level, the software performs the following actions after you issue the **request system software in-service-upgrade *package-name*** command to upgrade to a new Junos OS software release in a two-member Virtual Chassis configuration:

1. Arms the new Junos OS software release on all Routing Engines in the Virtual Chassis.

   The Routing Engines are still running the old Junos OS software release.

2. Upgrades both standby (backup) Routing Engines (VC-Ms and VC-Bs) in the Virtual Chassis.

   The Virtual Chassis is still actively forwarding traffic.

3. Performs a local switchover of the Routing Engines in the Virtual Chassis backup router (VC-B).

The local switchover causes the VC-Bs upgraded in Step 2 to become the VC-Bm, and the VC-Bm that was still running the old Junos OS software to become the VC-Bs. The VC-Bm is now running the new Junos OS software release, and the VC-Bs is still running the old Junos OS software release. The Virtual Chassis is still actively forwarding traffic.

4. Upgrades the Packet Forwarding Engines to the new Junos OS software release.

   The Packet Forwarding Engines are now using the upgraded VC-Bm as the Virtual Chassis protocol master.

5. Performs a local switchover of the Routing Engines in the Virtual Chassis master router (VC-M).

   The local switchover of the VC-M also causes a global switchover in the Virtual Chassis, which causes the VC-M to become the VC-B. As a result, the VC-Mm becomes the VC-Bs, and the VC-Ms becomes the VC-Bm. The global switchover on the VC-B causes the VC-Bm to become the VC-Mm, and the VC-Bs to become the VC-Ms.

   The VC-Mm and VC-Bm are now running the new Junos OS software release. The VC-Ms (originally the VC-Bm) and VC-Bs (originally the VC-Mm) are still running the old Junos OS software release.

6. Upgrades the standby Routing Engines in the Virtual Chassis (VC-Ms and VC-Bs).

   The Virtual Chassis is now fully upgraded to the new Junos OS software release.

Virtual Chassis Role Transitions After a Unified ISSU

A unified ISSU in an MX Series Virtual Chassis upgrades all Routing Engines in the Virtual Chassis to the new Junos OS software release. In a two-member Virtual Chassis, this includes four Routing Engines: the master and standby (backup) Routing Engines in the Virtual Chassis master router, and the master and standby Routing Engines in the Virtual Chassis backup router. As a result, the member routers and their associated Routing Engines undergo both global and local role transitions after the unified ISSU completes.

A *global role transition* changes the mastership in the Virtual Chassis by switching the global roles of the Virtual Chassis master router (VC-M) and Virtual Chassis backup router (VC-B), and applies globally across the entire Virtual Chassis. A *local role transition* toggles the local mastership roles (**master** and **standby**, or **m** and **s**) of each of the two Routing Engines in a member router, and applies locally only to that member router.

A unified ISSU in an MX Series Virtual Chassis causes the global and local role transitions listed in Table 21 on page 173.

Table 21: Virtual Chassis Role Transitions After Unified ISSU

Virtual Chassis Role *Before* Unified ISSU	Virtual Chassis Role *After* Unified ISSU	Type of Role Change
Virtual Chassis master router (VC-M)	Virtual Chassis backup router (VC-B)	Global

Table 21: Virtual Chassis Role Transitions After Unified ISSU *(continued)*

Virtual Chassis Role *Before* Unified ISSU	Virtual Chassis Role *After* Unified ISSU	Type of Role Change
Virtual Chassis backup router (VC-B)	Virtual Chassis master router (VC-M)	Global
Master Routing Engine in the Virtual Chassis master router (VC-Mm)	Standby Routing Engine in the Virtual Chassis backup router (VC-Bs)	Local
Standby Routing Engine in the Virtual Chassis master router (VC-Ms)	Master Routing Engine in the Virtual Chassis backup router (VC-Bm)	Local
Master Routing Engine in the Virtual Chassis backup router (VC-Bm)	Standby Routing Engine in the Virtual Chassis master router (VC-Ms)	Local
Standby Routing Engine in the Virtual Chassis backup router (VC-Bs)	Master Routing Engine in the Virtual Chassis master router(VC-Mm)	Local

Release History Table

Release	Description
14.1	Starting in Junos OS Release 14.1, you can perform a unified in-service software upgrade (unified ISSU) for an MX Series Virtual Chassis configuration.

Related Documentation

- Preparing for a Unified ISSU in an MX Series Virtual Chassis on page 175
- Upgrading Junos OS in an MX Series Virtual Chassis by Performing a Unified ISSU on page 177
- Global Roles and Local Roles in a Virtual Chassis on page 30
- *Unified ISSU System Requirements*

Preparing for a Unified ISSU in an MX Series Virtual Chassis

Starting in Junos OS Release 14.1, you can perform a unified in-service software upgrade (unified ISSU) in an MX Series Virtual Chassis. Before you begin such an upgrade, perform the following tasks to help ensure its success.

NOTE: For recommended settings on MX Virtual Chassis devices, please consult Maximizing Scaling and Performance for MX Series Virtual Chassis on the Juniper Networks Knowledge Base.

To prepare for a unified ISSU in a two-member MX Series Virtual Chassis:

1. Back up the existing system software to each member router's hard disk so you can roll back to it if necessary.

   Isssue the **request system snapshot all-members** command to archive data and executable areas for all members of the Virtual Chassis configuration.

2. Make sure enhanced IP network services is configured.

   See "Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis" on page 75.

3. Verify that both graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) are enabled.

   See "Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis" on page 75.

4. Make sure the router is configured to delay for 180 seconds (3 minutes) before removing access routes and access-internal routes for DHCP and PPP subscriber management after a graceful Routing Engine switchover takes place.

   See *Delaying Removal of Access Routes and Access-Internal Routes After Graceful Routing Engine Switchover*.

5. Ensure that the router is configured to use a *hold-time* value of 300 seconds when negotiating a BGP connection with the peer.

   See *BGP Messages Overview*.

6. Stop all CPU-intensive periodic operations running on the Virtual Chassis in order to conserve system resources.

   Examples of such CPU-intensive operations include, but are not limited to, periodic SNMP Get and SNMP Walk requests, isssuing Junos OS operational mode commands with detailed or extensive output, and runnning Stylesheet Language Alternative Syntax (SLAX) operational scripts that monitor device status or network events.

7. (Aggregated Ethernet interfaces only) If you are using aggregated Ethernet interfaces, make sure each interface is configured to use a **slow** interval (every 30 seconds) for *periodic* transmission of Link Aggregation Control Protocol (LACP) packets.

   See *Configuring LACP for Aggregated Ethernet Interfaces*.

8. (IS-IS interfaces only) If you are using IS-IS interfaces configured with the **iso** protocol family, make sure the link-state PDU lifetime value (*lsp-lifetime*) is set to 65317 seconds.

   See *Example: Configuring the Transmission Frequency for Link-State PDUs on IS-IS Interfaces*.

BEST PRACTICE: **For additional information about unified ISSU, consult our Knowledge Base.**

Release History Table

Release	Description
14.1	Starting in Junos OS Release 14.1, you can perform a unified in-service software upgrade (unified ISSU) in an MX Series Virtual Chassis.

Related Documentation

- Upgrading Junos OS in an MX Series Virtual Chassis by Performing a Unified ISSU on page 177
- Unified ISSU in a Virtual Chassis on page 171

Upgrading Junos OS in an MX Series Virtual Chassis by Performing a Unified ISSU

Starting in Junos OS Release 14.1, you can upgrade the member routers in a two-member MX Series Virtual Chassis to a later release by performing a unified in-service software upgrade (unified ISSU). Unified ISSU enables you to upgrade to a new Junos OS software release with minimal traffic disruption and no loss of subscriber sessions.

The unified ISSU procedure upgrades and changes the roles of all Routing Engines in the Virtual Chassis to the new Junos OS software release. In a two-member Virtual Chassis, this includes four Routing Engines: the master and standby Routing Engines in the Virtual Chassis master router, and the master and standby Routing Engines in the Virtual Chassis backup router.

This procedure describes how to upgrade Junos OS in an MX Series Virtual Chassis by performing a unified ISSU. For information about upgrading Junos OS in an MX Series Virtual Chassis by rebooting the Routing Engines, see "Example: Upgrading Junos OS in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers by Rebooting the Routing Engines" on page 165.

Before you begin a unified ISSU in a two-member MX Series Virtual Chassis, perform the following tasks:

- Prepare the member routers for the unified ISSU operation.

  See "Preparing for a Unified ISSU in an MX Series Virtual Chassis" on page 175.

To perform a unified ISSU in a two-member MX Series Virtual Chassis:

1. Use FTP or a Web browser to download the Junos OS software package from the Juniper Networks Support Web site.

   See *Downloading Software*.

2. Open four console windows to access the console ports on each of the four Routing Engines in the Virtual Chassis:

   - Master Routing Engine in the Virtual Chassis master router (VC-Mm)
   - Standby Routing Engine in the Virtual Chassis master router (VC-Ms)
   - Master Routing Engine in the Virtual Chassis backup router (VC-Bm)
   - Standby Routing Engine in the Virtual Chassis backup router (VC-Bs)

   Opening separate console windows enables you to monitor the progress of the unified ISSU on each of the Routing Engines.

3. (Optional but recommended) Confirm that the member routers are ready for the unified ISSU process.

   Issue the following commands in each console window:

   ```
   {master:member0-re0}
   ```



```
user@host> show virtual-chassis vc-port | match “Configured”
user@host> show system switchover local
user@host> show system uptime local
```

Issuing these commands provides the following information:

- The **show virtual-chassis vc-port | match “Configured”** command output confirms that all Virtual Chassis port interfaces are properly configured and operational.
- The **show system switchover local** command output confirms that the configuration database and kernel database are ready for a unified ISSU.
- The **show system uptime local** command output displays the date and time when this Routing Engine was last booted, and how long it has been running. It typically takes from 5 minutes to 15 minutes since the last switchover or system reboot for the Routing Engine to be ready for a unified ISSU.

4. Verify that all four Routing Engines in the Virtual Chassis are running the same Junos OS software release.

   Issue the **show version invoke-on all-routing-engines** command with the **all-members** option to display the hostnames and version information about the software running on all Routing Engines in the Virtual Chassis.

   ```
   {master:member0-re0}
   user@host> show version all-members invoke-on all-routing-engines
   ```

5. Initiate the unified ISSU process.

   In the console window on the VC-Mm, issue the **request system software in-service-upgrade *package-name*** command.

   ```
   {master:member0-re0}
   user@host> request system software in-service-upgrade package-name
   ```

   NOTE: You do not need to specify the reboot option for the request system software in-service-upgrade command because a unified ISSU in an MX Series Virtual Chassis always reboots all Routing Engines in the member routers.

   For MX series routers with RE-S-X6-64G Routing Engine, upgrade the guest Junos OS and the Linux VM host OS by issuing the request vmhost software in-service-upgrade *package-name* command.

6. Check the console windows for each Routing Engine to monitor the unified ISSU progress and determine when the upgrade is complete.

   The unified ISSU can take between 30 minutes and 90 minutes to complete, depending on the size of your configuration. When the unified ISSU completes, the login prompt appears in the console windows.

For an example of the output for a unified ISSU operation in an MX Series Virtual Chassis, see **request system software in-service-upgrade** and *request vmhost software in-service-upgrade*.

Release History Table

Release	Description
14.1	Starting in Junos OS Release 14.1, you can upgrade the member routers in a two-member MX Series Virtual Chassis to a later release by performing a unified in-service software upgrade (unified ISSU).

Related Documentation

- Preparing for a Unified ISSU in an MX Series Virtual Chassis on page 175
- Unified ISSU in a Virtual Chassis on page 171
- *Example: Performing a Unified ISSU*
- *Unified ISSU System Requirements*
- Example: Upgrading Junos OS in a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers by Rebooting the Routing Engines on page 165

CHAPTER 10

Monitoring an MX Series Virtual Chassis

- Accessing the Virtual Chassis Through the Management Interface on page 182
- Verifying the Status of Virtual Chassis Member Routers or Switches on page 183
- Verifying the Operation of Virtual Chassis Ports on page 183
- Verifying Neighbor Reachability for Member Routers or Switches in a Virtual Chassis on page 184
- Verifying Neighbor Reachability for Hardware Devices in a Virtual Chassis on page 184
- Determining GRES Readiness in a Virtual Chassis Configuration on page 185
- Viewing Information in the Virtual Chassis Control Protocol Adjacency Database on page 186
- Viewing Information in the Virtual Chassis Control Protocol Link-State Database on page 187
- Viewing Information About Virtual Chassis Port Interfaces in the Virtual Chassis Control Protocol Database on page 188
- Viewing Virtual Chassis Control Protocol Routing Tables on page 188
- Viewing Virtual Chassis Control Protocol Statistics for Member Devices and Virtual Chassis Ports on page 189
- Verifying and Managing the Virtual Chassis Heartbeat Connection on page 190
- Inline Flow Monitoring for Virtual Chassis Overview on page 191
- Managing Files on Virtual Chassis Member Routers or Switches on page 193
- Virtual Chassis SNMP Traps on page 194
- Virtual Chassis Slot Number Mapping for Use with SNMP on page 194
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet on page 196
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets on page 207

Accessing the Virtual Chassis Through the Management Interface

The management Ethernet interface (**fxp0**) on an MX Series router or EX9200 switch is an out-of-band management interface, also referred to as a management port, that enables you to use Telnet or SSH to access and manage the device remotely. You typically configure the management interface with an IP address and prefix length when you first install Junos OS.

You can configure a management Ethernet interface in one of two ways to access a Virtual Chassis:

- To access the Virtual Chassis as a whole, configure a consistent IP address for the management interface using the **master-only** option. You can use this management IP address to consistently access the master (primary) Routing Engine in the master router or switch (protocol master) for the Virtual Chassis.
- To access a specific Routing Engine in an individual member router or switch of the Virtual Chassis, configure an IP address for one of the following configuration groups:
  - **member0-re0**
  - **member0-re1**
  - **member1-re0**
  - **member1-re1**

BEST PRACTICE: For most management tasks, we recommend that you access the Virtual Chassis as a whole through a consistent management IP address. For troubleshooting purposes, however, accessing a specific Routing Engine in an individual member router or switch may be useful.

To access a Virtual Chassis through the management Ethernet interface, do one of the following:

- Configure a consistent management IP address that accesses the entire Virtual Chassis through the master Routing Engine in the Virtual Chassis master router or switch.

  ```
  {master:member0-re0}[edit]
  user@host# set interfaces fxp0 unit 0 family inet address ip-address/prefix-length
    master-only
  ```

  For example, to access the entire Virtual Chassis via management IP address **10.4.5.33/16**:

  ```
  {master:member0-re0}[edit]
  user@host# set interfaces fxp0 unit 0 family inet address 10.4.5.33/16 master-only
  ```

- Configure a management IP address that accesses a specified Routing Engine in an individual member router or switch in the Virtual Chassis.

  ```
  {master:member0-re0}[edit groups]
  ```

```
user@host# set membern-ren interfaces fxp0 unit 0 family inet address
  ip-address/prefix-length
```

For example, to access the Routing Engine installed in slot 1 of member router 1 (**member1-re1**) in the Virtual Chassis:

```
{master:member0-re0}[edit groups]
user@host# set member1-re1 interfaces fxp0 unit 0 family inet address 10.4.3.145/32
```

Related Documentation

- *Configuring a Consistent Management IP Address*

Verifying the Status of Virtual Chassis Member Routers or Switches

Purpose Verify that the member routers or switches in an MX Series or EX9200 Virtual Chassis are properly configured.

Action Display the status of the members of the Virtual Chassis configuration:

```
user@host> show virtual-chassis status
```

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- *Configuring an EX9200 Virtual Chassis*
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Verifying the Operation of Virtual Chassis Ports

Purpose Verify that the Virtual Chassis ports in an MX Series or EX9200 Virtual Chassis are properly configured and operational.

Action

- To display the status of the Virtual Chassis ports for both member routers or switches in the Virtual Chassis:

  ```
  user@host> show virtual-chassis vc-port all-members
  ```

- To display the status of the Virtual Chassis ports for a specified member router or switch in the Virtual Chassis:

  ```
  user@host> show virtual-chassis vc-port member member-id
  ```

- To display the status of the Virtual Chassis ports for the member router or switch on which you are issuing the command:

  ```
  user@host> show virtual-chassis vc-port local
  ```

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- *Configuring an EX9200 Virtual Chassis*
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Verifying Neighbor Reachability for Member Routers or Switches in a Virtual Chassis

Purpose Verify that each member router or switch in an MX Series or EX9200 Virtual Chassis has a path to reach the neighbor devices to which it is connected.

Action

- To display neighbor reachability information for both member devices in the Virtual Chassis:

  ```
  user@host> show virtual-chassis active-topology all-members
  ```

- To display neighbor reachability information for a specified member device in the Virtual Chassis:

  ```
  user@host> show virtual-chassis active-topology member member-id
  ```

- To display neighbor reachability information for the member device on which you are issuing the command:

  ```
  user@host> show virtual-chassis active-topology local
  ```

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- *Configuring an EX9200 Virtual Chassis*
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Verifying Neighbor Reachability for Hardware Devices in a Virtual Chassis

Purpose Verify that each hardware device in an MX Series Virtual Chassis or an EX9200 Virtual Chassis can reach the neighbor routers and devices to which it is connected. On the MX Series routing platform, there is only one active device for each member router.

Action

- To display neighbor reachability information for the devices in both member routers in the Virtual Chassis:

  ```
  user@host> show virtual-chassis device-topology all-members
  ```

- To display neighbor reachability information for the device in a specified member router in the Virtual Chassis:

```
user@host> show virtual-chassis device-topology member member-id
```

- To display neighbor reachability information for the device in the member router on which you are issuing the command:

```
user@host> show virtual-chassis device-topology local
```

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- *Configuring an EX9200 Virtual Chassis*
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Determining GRES Readiness in a Virtual Chassis Configuration

Depending on the configuration, a variable amount of time is required before a router or switch is ready to perform a graceful Routing Engine switchover (GRES). Attempting a GRES operation before the device is ready can cause system errors and unexpected behavior.

To determine whether the member routers or switches in a Virtual Chassis configuration are ready for a GRES operation from a database synchronization perspective, you can issue the **request virtual-chassis routing-engine master switch check** command from the Virtual Chassis master router or switch (VC-Mm) before you initiate the GRES operation. Using the **request virtual-chassis routing-engine master switch check** command before you initiate the GRES operation ensures that the subscriber management and kernel databases on both member routers or switches are synchronized and ready for the GRES operation.

To determine whether the member routers or switches are ready for GRES from a database synchronization perspective:

1. Issue the **request virtual-chassis routing-engine master switch check** command from the Virtual Chassis master router or switch (VC-Mm).

   ```
   {master:member0-re0}
   user@host> request virtual-chassis routing-engine master switch check
   ```

   The **request virtual-chassis routing-engine master switch check** command checks various system and database components on the member routers or switches to determine whether they are ready for GRES, but does not initiate the global GRES operation itself. The readiness check includes ensuring that a system timer, which expires after 300 seconds, completes before the global GRES operation begins.

2. Review the results of the **request virtual-chassis routing-engine master switch check** command to determine whether the member routers or switches are ready for a GRES operation from a database synchronization perspective.

   - If the member routers or switches are ready for GRES, the **request virtual-chassis routing-engine master switch check** command displays a message confirming GRES readiness. For example:

     ```
     {master:member0-re0}

     user@host> request virtual-chassis routing-engine master switch check
     Switchover Ready
     ```

   - If the member routers or switches are not ready for GRES, the **request virtual-chassis routing-engine master switch check** command displays information about the readiness of the system. For example:

     ```
     {master:member0-re0}

     user@host> request virtual-chassis routing-engine master switch check
     error: chassisd Not ready for mastership switch, try after 217 secs.
     mastership switch request NOT honored, backup not ready
     ```

     The specific command output differs depending on the GRES readiness state of the member routers or switches.

Related Documentation

- Switchover Behavior in an MX Series Virtual Chassis on page 34
- Virtual Chassis Components Overview on page 24
- Global Roles and Local Roles in a Virtual Chassis on page 30
- *Understanding Graceful Routing Engine Switchover*

Viewing Information in the Virtual Chassis Control Protocol Adjacency Database

Purpose View information about neighbors in the Virtual Chassis Control Protocol (VCCP) adjacency database for a Virtual Chassis configuration.

Action

- To display VCCP neighbor adjacency information for both member devices in the Virtual Chassis:

  ```
  user@host> show virtual-chassis protocol adjacency all-members
  ```

- To display VCCP neighbor adjacency information for a specified member device in the Virtual Chassis:

  ```
  user@host> show virtual-chassis protocol adjacency member member-id
  ```

- To display VCCP neighbor adjacency information for the device with a specified system ID:

  ```
  user@host> show virtual-chassis protocol adjacency system-id
  ```

- To display VCCP neighbor adjacency information for the device with a specified system ID on the specified member router or switch:

  ```
  user@host> show virtual-chassis protocol adjacency member member-id system-id
  ```

- To display VCCP neighbor adjacency information for the member device on which you are issuing the command:

  ```
  user@host> show virtual-chassis protocol adjacency local
  ```

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- *Configuring an EX9200 Virtual Chassis*
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Viewing Information in the Virtual Chassis Control Protocol Link-State Database

Purpose View information about protocol data unit (PDU) packets in the Virtual Chassis Control Protocol (VCCP) link-state database for a Virtual Chassis configuration.

Action

- To display VCCP PDU information for both member routers or switches in the Virtual Chassis:

  ```
  user@host> show virtual-chassis protocol database all-members
  ```

- To display VCCP PDU information for a specified member router or switch in the Virtual Chassis:

  ```
  user@host> show virtual-chassis protocol database member member-id
  ```

- To display VCCP PDU information for the device with a specified system ID:

  ```
  user@host> show virtual-chassis protocol database system-id
  ```

- To display VCCP PDU information for the device with a specified system ID on the specified member router or switch:

  ```
  user@host> show virtual-chassis protocol database member member-id system-id
  ```

- To display VCCP PDU information for the member router or switch on which you are issuing the command:

  ```
  user@host> show virtual-chassis protocol database local
  ```

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58

- *Configuring an EX9200 Virtual Chassis*
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Viewing Information About Virtual Chassis Port Interfaces in the Virtual Chassis Control Protocol Database

Purpose View information in the Virtual Chassis Control Protocol (VCCP) database about Virtual Chassis port interfaces in the Virtual Chassis.

Action

- To display VCCP information about Virtual Chassis port interfaces for both member routers or switches:

  ```
  user@host> show virtual-chassis protocol interface all-members
  ```

- To display VCCP information about Virtual Chassis port interfaces for a specified member router or switch:

  ```
  user@host> show virtual-chassis protocol interface member member-id
  ```

- To display VCCP information about a specified Virtual Chassis port interface:

  ```
  user@host> show virtual-chassis protocol interface vcp-slot/pic/port.logical-unit-number
  ```

- To display VCCP information about Virtual Chassis port interfaces for the member router or switch on which you are issuing the command:

  ```
  user@host> show virtual-chassis protocol interface local
  ```

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- *Configuring an EX9200 Virtual Chassis*
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Viewing Virtual Chassis Control Protocol Routing Tables

Purpose View Virtual Chassis Control Protocol (VCCP) unicast and multicast routing tables for an MX Series Virtual Chassis configuration.

Action

- To display the VCCP unicast and multicast routing tables for both member routers in the Virtual Chassis:

  ```
  user@host> show virtual-chassis protocol route all-members
  ```

- To display the VCCP unicast and multicast routing tables for a specified member router in the Virtual Chassis:

  ```
  user@host> show virtual-chassis protocol route member member-id
  ```

- To display the VCCP unicast and multicast routing tables to the destination with the specified system ID:

  ```
  user@host> show virtual-chassis protocol route destination-id
  ```

- To display the VCCP unicast and multicast routing tables to the destination with the specified system ID on the specified member router:

  ```
  user@host> show virtual-chassis protocol route member member-id destination-id
  ```

- To display the VCCP unicast and multicast routing tables for the member router on which you are issuing the command:

  ```
  user@host> show virtual-chassis protocol route local
  ```

Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Viewing Virtual Chassis Control Protocol Statistics for Member Devices and Virtual Chassis Ports

Purpose View Virtual Chassis Control Protocol (VCCP) statistics for one or both member routers or switches, or for a specified Virtual Chassis port interface, in a Virtual Chassis configuration.

Action

- To display VCCP statistics for both member routers or switches in the Virtual Chassis:

  ```
  user@host> show virtual-chassis protocol statistics all-members
  ```

- To display VCCP statistics for a specified member router or switch in the Virtual Chassis:

  ```
  user@host> show virtual-chassis protocol statistics member member-id
  ```

- To display VCCP statistics for a specified Virtual Chassis port interface:

  ```
  user@host> show virtual-chassis protocol statistics vcp-slot/pic/port.logical-unit-number
  ```

- To display VCCP statistics for the member router or switch on which you are issuing the command:

  ```
  user@host> show virtual-chassis protocol statistics local
  ```


Related Documentation

- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- *Configuring an EX9200 Virtual Chassis*
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Verifying and Managing the Virtual Chassis Heartbeat Connection

Purpose Verify the configuration and operation of a heartbeat connection for an MX Series Virtual Chassis.

NOTE: Starting in Junos OS Release 14.1, you can configure a heartbeat connection for an MX Series Virtual Chassis.

Action

- To clear heartbeat connection statistics counters and timestamp fields for all (both) member routers, the specified member router, or the member router on which you are issuing the command:

```
user@host> clear virtual-chassis heartbeat <all-members>
user@host> clear virtual-chassis heartbeat member member-id
user@host> clear virtual-chassis heartbeat local
```

- To view the heartbeat connection state for all (both) member routers, the specified member router, or the member router on which you are issuing the command:

```
user@host> show virtual-chassis heartbeat <all-members>
user@host> show virtual-chassis heartbeat member member-id
user@host> show virtual-chassis heartbeat local
```

- To display and review the statistics collected by the heartbeat connection for all (both) member routers, the specified member router, or the member router on which you are issuing the command:

```
user@host> show virtual-chassis heartbeat detail <all-members>
user@host> show virtual-chassis heartbeat detail member member-id
user@host> show virtual-chassis heartbeat detail local
```

- To verify use (status) of the heartbeat connection when an adjacency disruption or split is detected in the Virtual Chassis:

```
user@host> show virtual-chassis status
```

Release History Table

Release	Description
14.1	Starting in Junos OS Release 14.1, you can configure a heartbeat connection for an MX Series Virtual Chassis.

Related Documentation

- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet on page 196
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets on page 207
- Virtual Chassis Heartbeat Connection Overview on page 41

Inline Flow Monitoring for Virtual Chassis Overview

Inline flow monitoring enables you to monitor the flow of traffic by means of a router or switch participating in a network.

Inline flow monitoring for an MX Series Virtual Chassis or an EX9200 Virtual Chassis supports the following features:

- Active sampling and exporting of both IPv4 and IPv6 traffic flows. Active (inline) sampling occurs on an inline data path without the need for a services DPC.
- Sampling traffic flows in both the ingress and egress directions.
- Configuring flow collection on either IPv4 or IPv6 devices.
- Using the IPFIX flow collection template for traffic sampling. The IPFIX template supports both IPv4 and IPv6 export records.
- Sampling and exporting of VPLS flows
- Exporting of data in Version-IPFIX and Version 9 formats

Consider the following guidelines when you configure Virtual Chassis for inline flow monitoring.

- Split Detection on page 191
- Syntax of the sampling-instance Statement on page 192
- FPC Slot Numbers for the Virtual Chassis on page 192

Split Detection

In a two-member MX Series Virtual Chassis or an EX9200 Virtual Chassis , you must disable split detection to configure and use inline flow monitoring. To do so, include the **no-split-detection** statement at the **[edit virtual-chassis]** hierarchy level when you set up the Virtual Chassis.

Syntax of the sampling-instance Statement

To associate a sampling instance with an FPC in the Virtual Chassis master router (member ID 0), use the **sampling-instance *instance-name*** statement at the **[edit chassis member *member-number* fpc slot *slot-number*]** hierarchy level, where ***member-number*** is 0 (zero) and ***slot-number*** is a number in the range 0 through 11. For example, the following statement associates a sampling instance named sample1 to the FPC in slot 1 of a Virtual Chassis master router:

```
[edit chassis member 0 fpc slot 1]
user@host# set sampling-instance sample1
```

To associate a sampling instance with an FPC in the Virtual Chassis backup router (member ID 1), use the **sampling-instance *instance-name*** statement at the **[edit chassis member *member-number* fpc slot *slot-number*]** hierarchy level, where ***member-number*** is 1 and ***slot-number*** is a number in the range 0 through 11. For example, the following statement associates a sampling instance named sample2 to the FPC in slot 2 of Virtual Chassis backup router:

```
[edit chassis member 1 fpc slot 2]
user@host# set sampling-instance sample2
```

FPC Slot Numbers for the Virtual Chassis

After you configure the member ID and, optionally, slot count for each router that you want to add to the Virtual Chassis, the Routing Engines in that chassis reboot and the slots for line cards (FPCs) are renumbered. The FPC slot numbering used for each member router is based on the slot count and offsets used in the Virtual Chassis instead of the physical slot numbers where the line card is actually installed.

For example, assume that in your Virtual Chassis configuration, member 0 is an MX960 router and member 1 is an MX2010 router, with the default slot count (12) in effect on both routers. In this topology, a 10-Gigabit Ethernet interface that appears as xe-14/2/2 (FPC slot 14, PIC slot 2, port 2) in the **show services accounting status inline-jflow** command output or the **show interfaces** command output is actually physical interface xe-2/2/2 (FPC slot 2, PIC slot 2, port 2) on member 1 after deducting the offset of 12 for member 1.

NOTE: **Platform support and associated slots depend on the Junos OS release in your installation.**

For more information about how slot count and slot numbering work in an MX Series Virtual Chassis, see "Virtual Chassis Components Overview" on page 24.

Related Documentation

- *Configuring Inline Active Flow Monitoring*
- *Sampling Instance Configuration*
- no-split-detection on page 241
- Disabling Split Detection in a Virtual Chassis Configuration on page 98
- Split Detection Behavior in a Virtual Chassis on page 38

Managing Files on Virtual Chassis Member Routers or Switches

In a Virtual Chassis configuration for MX Series 3D Universal Edge Routers or EX9200 switches, you can manage files on local and remote member routers or switches by including a member specification in the following **file** operational commands:

file archive	**file copy**
file checksum md5	**file delete**
file checksum sha1	**file list**
file checksum sha-256	**file rename**
file compare	**file show**

The member specification identifies the specific Virtual Chassis member router or switch and Routing Engine on which you want to manage files, and includes both of the following elements:

- The Virtual Chassis member ID (**0** or **1**)
- The Routing Engine slot number (**re0** or **re1**)

To manage files on a specific member router or switch and a specific Routing Engine:

- From operational mode, issue the **file** command and Virtual Chassis member specification:

```
{master:member0-re0}
user@host> file option member(0 | 1)-re(0 | 1):command-option
```

For example, the following **file list** command uses the **member1-re0** specification to display a list of the files in the **/config** directory on the Routing Engine in slot 0 (**re0**) in Virtual Chassis **member 1**. The router or switch forwards the command from **member 0**, where it is issued, to **member 1**, and displays the results as if the command were processed on the local device.

```
{master:member0-re0}
user@host> file list member1-re0:/config
member1-re0:
--------------------------------------------------------------------------

/config:
.snap/
juniper.conf.1.gz
juniper.conf.2.gz
juniper.conf.3.gz
juniper.conf.gz
juniper.conf.md5
license/
license.old/
```

```
usage.db
vchassis/
```

Related Documentation

- Interchassis Redundancy and Virtual Chassis Overview on page 19
- Virtual Chassis Components Overview on page 24
- *Format for Specifying Filenames and URLs in Junos OS CLI Commands*

Virtual Chassis SNMP Traps

Junos OS supports the use of SNMP traps to monitor the routers, switches, and other devices in your network.

MX Virtual Chassis supports the following enterprise-specific traps:

- **jnxVccpPortUp**
- **jnxVccpPortDown**

An unexpected SNMP trap of **jnxVccpPortDown** with a **jnxVirtualChassisPortOperStatus** value of 2 (down) should be treated as a critical notification, as it indicates one or more of the VCP-interfaces has undergone a failure, and may result in degraded performance or decreased resiliency to further errors until the status is cleared by a **jnxVccpPortUp** with **jnxVirtualChassisPortOperStatus** value equal to 1 (up).

A detailed description may be found in the MIB, located http://www.juniper.net/techpubs/en_US/junos15.1/topics/reference/mibs/mib-jnx-virtualchassis.txt.

Related Documentation

- Virtual Chassis Slot Number Mapping for Use with SNMP on page 194
- Virtual Chassis Components Overview on page 24
- SNMP MIB Explorer

Virtual Chassis Slot Number Mapping for Use with SNMP

Junos OS supports the use of SNMP to monitor the routers, switches, and other devices in your network. For example, the Juniper Networks jnxBoxAnatomy enterprise-specific Chassis MIB contains the jnxFruTable object, which shows the status of field-replaceable units (FRUs) in the chassis. Within the jnxFruTable object, the jnxFruSlot object displays the slot number where the FRU is installed.

If you are using the jnxFruSlot object in jnxFruTable to display the slot numbers of line cards installed in a member router of an MX Series Virtual Chassis or a member switch of an EX9200 Virtual Chassis, keep in mind that the offset used for slot numbering in the Virtual Chassis affects the value that appears for the jnxFruSlot object.

Table 22 on page 195 lists the jnxFruSlot number that appears in the jnxFruTable of the jnxBoxAnatomy MIB, and the corresponding line card physical slot number in each member router of a two-member EX9200 or MX Series Virtual Chassis. For example, a jnxFruSlot

value of 15 corresponds to physical slot 3 in member 0 of the Virtual Chassis. A jnxFruSlot value of 30 corresponds to physical slot 6 in member 1 of the Virtual Chassis.

Table 22: jnxFruSlot Numbers and Corresponding Slot Numbers in an MX Series or EX9200 Virtual Chassis

jnxFruSlot Number	Line Card Slot Number	MX Series or EX9200 Virtual Chassis Member ID
Line Cards in MX Series Virtual Chassis Member ID 0 (offset = 12):		
12	0	0
13	1	0
14	2	0
15	3	0
16	4	0
17	5	0
18	6	0
19	7	0
20	8	0
21	9	0
22	10	0
23	11	0
Line Cards in MX Series Virtual Chassis Member ID 1 (offset = 24)		
24	0	1
25	1	1
26	2	1
27	3	1
28	4	1
29	5	1
30	6	1

Table 22: jnxFruSlot Numbers and Corresponding Slot Numbers in an MX Series or EX9200 Virtual Chassis *(continued)*

jnxFruSlot Number	Line Card Slot Number	MX Series or EX9200 Virtual Chassis Member ID
31	7	1
32	8	1
33	9	1
34	10	1
35	11	1

Related Documentation

- Virtual Chassis Components Overview on page 24
- SNMP MIB Explorer

Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet

A *heartbeat connection* is an IP-based, bidirectional packet connection in an MX Series Virtual Chassis between the Virtual Chassis master and backup routers. The *heartbeat packets* exchanged over this connection provide critical information about the availability and health of each member router. Starting in Junos OS Release 14.1, you can configure a heartbeat connection in an MX Series Virtual Chassis.

This example describes how to configure a heartbeat connection in an MX Series Virtual Chassis when both member routers reside in the same subnet. For information about configuring a heartbeat connection when the Virtual Chassis member routers reside in different subnets, see "Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets" on page 207.

- Requirements on page 196
- Overview on page 197
- Configuration on page 199
- Verification on page 204

Requirements

This example uses the following software and hardware components:

- Junos OS Release 14.1 and later releases
- Two MX240 3D Universal Edge Routers, each with dual Routing Engines

This configuration example has been tested using the software release listed and is assumed to work on all later releases.

BEST PRACTICE: We recommend that you use the **commit synchronize** command to save any configuration changes to the Virtual Chassis. For an MX Series Virtual Chassis, the **force** option is the default and only behavior when you issue the **commit synchronize** command. Issuing the **commit synchronize** command for an MX Series Virtual Chassis configuration has the same effect as issuing the **commit synchronize force** command.

Before you configure a heartbeat connection for a Virtual Chassis:

- Configure a Virtual Chassis consisting of two MX Series routers.

  See "Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis" on page 75

  As part of the preprovisioned Virtual Chassis configuration shown in the configuration example, you must create and apply the **member0-re0**, **member0-re1**, **member1-re0**, and **member1-re1** configuration groups for each member Routing Engine. Each configuration group includes a unique IP address for the management Ethernet interface (**fxp0**) on each Routing Engine.

  NOTE: When you create the preprovisioned Virtual Chassis configuration at the **[edit virtual-chassis]** hierarchy level, make sure you do *not* configure the **no-split-detection** statement to disable detection of a split in the Virtual Chassis. Using the **no-split-detection** statement is prohibited when you configure a Virtual Chassis heartbeat connection, and doing so causes the commit operation to fail.

- Ensure TCP connectivity between the master Routing Engine in the Virtual Chassis master router (VC-Mm) and the master Routing Engine in the Virtual Chassis backup router (VC-Bm).

  The Virtual Chassis heartbeat connection opens a proprietary TCP port numbered 33087 on the VC-Mm to listen for heartbeat messages. If your network design includes firewalls or filters, make sure the network allows traffic between TCP port 33087 on the VC-Mm and the dynamically allocated TCP port on the VC-Bm.

Overview

A *heartbeat connection* is an IP-based, bidirectional packet connection between the master router and backup router in an MX Series Virtual Chassis. The member routers forming the heartbeat connection exchange *heartbeat packets* that provide critical information about the availability and health of each member router. During a disruption or split in the Virtual Chassis configuration, the heartbeat connection prevents the member routers from changing mastership roles unnecessarily, which can cause undesirable results.

This example configures a heartbeat connection for an MX Series Virtual Chassis in which both MX240 member routers reside in the 10.4.0.0 subnet. Member router **caelum** is the

global master router for the Virtual Chassis (VC-M), and member router **elnath** is the global backup router (VC-B). Both member routers have dual Routing Engines installed, and the heartbeat connection is configured between the master Routing Engine in **caelum** (represented by VC-Mm or **member0-re0**) and the master Routing Engine in **elnath** (represented by VC-Bm or **member1-re0**).

Configuring a heartbeat connection for an MX Series Virtual Chassis when both member routers reside in the same subnet consists of the following tasks:

1. Configure the global **master-only** IP address for the **fxp0** management interface on the same subnet as the four Routing Engines in the Virtual Chassis.
2. Configure a network path for the heartbeat connection.

   This example uses a global static route to provide a path for member routers in the same subnet to reach each other via a TCP/IP connection.
3. Configure the global **master-only** IP address for the **fxp0** management interface as the heartbeat address to establish the Virtual Chassis heartbeat connection.
4. (Optional) Configure a nondefault value for the Virtual Chassis heartbeat timeout interval.

Topology

This example configures a heartbeat connection for an MX Series Virtual Chassis with both member routers in the same subnet. For redundancy, each member router is configured with two Virtual Chassis ports.

Table 23 on page 198 shows the hardware and software configuration settings for each MX Series router in the Virtual Chassis.

Table 23: Components of the Sample MX Series Virtual Chassis with Member Routers in Same Subnet

Router Name	Hardware	Serial Number	Member ID	Role	Virtual Chassis Ports	Subnet
caelum	MX240 router with: • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member0-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member0-re1**)	JN11026AFAFC	0	routing-engine (master)	vcp-1/0/0 vcp-1/1/0	10.4.0.0/16

Table 23: Components of the Sample MX Series Virtual Chassis with Member Routers in Same Subnet *(continued)*

Router Name	Hardware	Serial Number	Member ID	Role	Virtual Chassis Ports	Subnet
elnath	MX240 router with: • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member1-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member1-re1**)	JN112C2FCAFC	1	routing-engine (backup)	vcp-2/0/0 vcp-2/1/0	10.4.0.0/16

Configuration

To configure a heartbeat connection in an MX Series Virtual Chassis with both member routers in the same subnet, perform these tasks:

- Configuring a Consistent Management IP Address for Each Routing Engine on page 200
- Configuring a Static Route Between the Master and Backup Member Routers on page 202
- Configuring the Heartbeat Address and Heartbeat Timeout on page 203

CLI Quick Configuration

To quickly configure a heartbeat connection for a Virtual Chassis with both member routers in the same subnet, copy the following commands and paste them into the router terminal window:

```
[edit]
set groups member0-re0 interfaces fxp0 unit 0 family inet address 10.4.2.210/16
    master-only
set groups member0-re1 interfaces fxp0 unit 0 family inet address 10.4.2.210/16
    master-only
set groups member1-re0 interfaces fxp0 unit 0 family inet address 10.4.2.210/16
    master-only
set groups member1-re1 interfaces fxp0 unit 0 family inet address 10.4.2.210/16
    master-only
set groups global routing-options static route 10.4.0.0/16 next-hop 10.4.0.1
set groups global routing-options static route 10.4.0.0/16 retain
set groups global routing-options static route 10.4.0.0/16 no-readvertise
set virtual-chassis heartbeat-address 10.4.2.210
set virtual-chassis heartbeat-timeout 20
```

Configuring a Consistent Management IP Address for Each Routing Engine

Step-by-Step Procedure

In addition to configuring a unique IP address for the **fxp0** management interface on each Routing Engine when you first set up the Virtual Chassis, you must configure an additional management IP address, known as the **master-only** address, to ensure consistent access to the **fxp0** management interface on the master Routing Engine in the Virtual Chassis master router (VC-Mm, represented by **member0-re0** in this example). You then use the **master-only** address as the heartbeat address to establish the Virtual Chassis heartbeat connection.

Because the Virtual Chassis member routers in this example both reside in the same subnet (10.4.0.0), you can configure the same **master-only** address for each Routing Engine. The **master-only** address is active only on the management interface for the VC-Mm. During a switchover, the **master-only** address moves to the new Routing Engine currently functioning as the VC-Mm.

To configure the **master-only fxp0** IP address for each Routing Engine:

- From the console on member 0, configure the same IP address for the **fxp0** management interface on each Routing Engine.

```
{master:member0-re0}[edit]
user@caelum# set groups member0-re0 interfaces fxp0 unit 0 family inet address
  10.4.2.210/16 master-only
user@caelum# set groups member0-re1 interfaces fxp0 unit 0 family inet address
  10.4.2.210/16 master-only
user@caelum# set groups member1-re0 interfaces fxp0 unit 0 family inet address
  10.4.2.210/16 master-only
user@caelum# set groups member1-re1 interfaces fxp0 unit 0 family inet address
  10.4.2.210/16 master-only
```

Results

From the console on the Virtual Chassis master router, display the results of the configuration for each configuration group. For brevity, portions of the configuration unrelated to this procedure are replaced by an ellipsis (...).

For **member0-re0**:

```
{master:member0-re0}[edit]
user@caelum# show groups member0-re0
system {
    host-name caelum;
    backup-router 10.4.0.1 destination [ 172.16.0.0/12 ... 10.204.0.0/16 ];
}
interfaces {
    fxp0 {
        unit 0 {
            family inet {
                address 10.4.2.100/16;
                address 10.4.2.210/16 {
                    master-only;
                }
            }
```

```
        }
    }
}
```

For **member0-re1**:

```
{master:member0-re0}[edit]
user@caelum# show groups member0-re1
system {
    host-name caelum1;
    backup-router 10.4.0.1 destination [ 172.16.0.0/12 ... 10.204.0.0/16 ];
}
interfaces {
    fxp0 {
        unit 0 {
            family inet {
                address 10.4.2.101/16;
                address 10.4.2.210/16 {
                    master-only;
                }
            }
        }
    }
}
```

For **member1-re0**:

```
{master:member0-re0}[edit]
user@caelum# show groups member1-re0
system {
    host-name elnath;
    backup-router 10.4.0.1 destination [ 172.16.0.0/12 ... 10.204.0.0/16 ];
}
interfaces {
    fxp0 {
        unit 0 {
            family inet {
                address 10.4.3.101/16;
                address 10.4.2.210/16 {
                    master-only;
                }
            }
        }
    }
}
```

For **member1-re1**:

```
{master:member0-re0}[edit]
user@caelum# show groups member1-re1
system {
    host-name elnath1;
    backup-router 10.4.0.1 destination [ 172.16.0.0/12 ... 10.204.0.0/16 ];
}
interfaces {
    fxp0 {
        unit 0 {
```

```
            family inet {
              address 10.4.3.102/16;
              address 10.4.2.210/16 {
                master-only;
              }
            }
          }
        }
      }
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring a Static Route Between the Master and Backup Member Routers

Step-by-Step Procedure

You must configure a secure and reliable path between the master router and backup router for the exchange of TCP/IP heartbeat packets. The heartbeat packets provide critical information about the availability and health of each member router.

The route you create for the heartbeat connection must be independent of the Virtual Chassis port links. Specifically, you must ensure that the master Routing Engine in the Virtual Chassis backup router (VC-Bm) can make a TCP/IP connection to the **master-only** IP address of the master Routing Engine in the Virtual Chassis master router (VC-Mm).

This examples creates a global static route between the member routers to configure the heartbeat path. However, you can choose the method that best meets your needs to configure the heartbeat path for member routers in the same subnet. For example, you might use the member router's default gateway for this purpose.

BEST PRACTICE: We recommend that you use the router management interface (**fxp0**) as the heartbeat path. The management interface is generally available earlier than the line card interfaces, and is typically connected to a more secure network than the other interfaces.

To create a static route between the master and backup member routers:

- From the console on member 0, configure a static route between the member routers in subnet 10.4.0.0.

```
{master:member0-re0}[edit]
user@caelum# set groups global routing-options static route 10.4.0.0/16 next-hop
  10.4.0.1
user@caelum# set groups global routing-options static route 10.4.0.0/16 retain
user@caelum# set groups global routing-options static route 10.4.0.0/16
  no-readvertise
```

Results

Display the results of the configuration. For brevity, portions of the configuration unrelated to this procedure are replaced by an ellipsis (...).

```
{master:member0-re0}[edit]
```

```
user@caelum# show groups global routing-options static
route 10.4.0.0/16 {
    next-hop 10.4.0.1;
    retain;
    no-readvertise;
}
...
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring the Heartbeat Address and Heartbeat Timeout

Step-by-Step Procedure

To establish the heartbeat connection in a two-member MX Series Virtual Chassis, you must configure the IP address for the connection between the master and backup member routers. To ensure consistent access to the master Routing Engine in the Virtual Chassis master router (VC-Mm) regardless of which Routing Engine is currently active, you set the heartbeat address to the previously configured global **master-only** IP address for the **fxp0** management interface.

Optionally, you can also configure a nondefault value for the heartbeat timeout interval. The heartbeat timeout is the maximum time within which a Virtual Chassis member router must respond to a heartbeat packet sent by the other member router. If you do not explicitly configure the heartbeat timeout interval, the default value (2 seconds) applies.

To configure the heartbeat address and heartbeat timeout:

1. From the console on member 0, specify that you want to edit the Virtual Chassis preprovisioned configuration.

   ```
   {master:member0-re0}[edit]
   user@caelum# edit virtual-chassis
   ```

2. Configure the common **master-only** IP address for the **fxp0** management interface as the heartbeat address.

   ```
   {master:member0-re0}[edit virtual-chassis]
   user@caelum# set heartbeat-address 10.4.2.210
   ```

3. (Optional) Configure a nondefault value for the heartbeat timeout interval.

   ```
   {master:member0-re0}[edit virtual-chassis]
   user@caelum# set heartbeat-timeout 20
   ```

Results

Display the results of the configuration.

```
{master:member0-re0}[edit]
user@caelum# show virtual-chassis
preprovisioned;
traceoptions {
    file VCCP size 100m;
    flag all;
```

```
}
heartbeat-address 10.4.2.210;
heartbeat-timeout 20;
member 0 {
  role routing-engine;
  serial-number JN11026AFAFC;
}
member 1 {
  role routing-engine;
  serial-number JN112C2FCAFC;
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

To confirm that the Virtual Chassis heartbeat connection is working properly, perform these tasks:

- Verifying the Virtual Chassis Heartbeat Connection on page 204
- Verifying Use of the Heartbeat Connection During an Adjacency Split or Disruption on page 205
- Verifying Virtual Chassis Member Health from Heartbeat Statistics on page 205

Verifying the Virtual Chassis Heartbeat Connection

Purpose Verify that the heartbeat connection between the Virtual Chassis member routers is properly configured and operational.

Action Display the state of one or both member routers when a heartbeat connection is configured.

```
{master:member0-re0}

user@caelum> show virtual-chassis heartbeat
member0:
--------------------------------------------------------------------------
Local        Remote          State        Time
10.4.2.210   10.4.3.101      Alive        2014-02-18 11:18:14 PST

member1:
--------------------------------------------------------------------------
Local        Remote          State        Time
10.4.3.101   10.4.2.210      Alive        2014-02-18 11:18:15 PST
```

Meaning For each member router, the command output displays the IP addresses of the local and remote member routers that form the heartbeat connection. The value **Alive** in the **State** field confirms that the master Routing Engine in the specified member router is connected and has received a heartbeat response message. The **Time** field specifies the date and time of the last connection state change.

Verifying Use of the Heartbeat Connection During an Adjacency Split or Disruption

Purpose Verify use of the heartbeat connection when an adjacency disruption or split is detected in the Virtual Chassis.

Action Display the status of the member routers in the Virtual Chassis:

```
{master:member0-re0}

user@caelum> show virtual-chassis status
Preprovisioned Virtual Chassis
Virtual Chassis ID: 4806.94d6.2362
                                                Mastership           Neighbor List
Member ID       Status   Serial No    Model    priority    Role      ID  Interface

0 (FPC   0- 11) Heartbt  JN11026AFAFC mx240          129  Master*    1  vcp-1/0/0
                                                                     1  vcp-1/1/0

1 (FPC  12- 23) Prsnt    JN112C2FCAFC mx240          129  Backup     0  vcp-2/0/0
                                                                     0  vcp-2/1/0
```

Meaning The **Status** field for member ID 0 displays **Heartbt**, which indicates that this member router has used the heartbeat packet connection to maintain mastership roles during an adjacency disruption or split in the Virtual Chassis configuration. The **Status** field for member ID 1 displays **Prsnt**, which indicates that this member router is connected to the Virtual Chassis.

If a router is not currently connected to the Virtual Chassis, the **Status** field displays **NotPrsnt**.

Verifying Virtual Chassis Member Health from Heartbeat Statistics

Purpose Use statistics collected by the heartbeat connection to verify the availability and health of each Virtual Chassis member router. You can also use the **show virtual-chassis heartbeat detail** command to determine the maximum latency and minimum latency in your network.

Action Display and review the statistics collected by the heartbeat connection.

```
{master:member0-re0}

user@caelum> show virtual-chassis heartbeat detail
member0:
--------------------------------------------------------------------------
Local           Remote          State        Time
10.4.2.210      10.4.3.101      Alive        2014-02-18 11:18:14 PST

Heartbeat statistics
  Heartbeats sent: 10079
  Heartbeats received: 10079
  Heartbeats lost/missed: 0
  Last time sent: 2014-02-18 20:03:10 PST (00:00:00 ago)
  Last time received: 2014-02-18 20:03:10 PST (00:00:00 ago)
  Maximum latency (secs): 0
  Minimum latency (secs): 0

member1:
--------------------------------------------------------------------------
Local           Remote          State        Time
10.4.3.101      10.4.2.210      Alive        2014-02-18 11:18:15 PST

Heartbeat statistics
  Heartbeats sent: 10083
  Heartbeats received: 10083
  Heartbeats lost/missed: 0
  Last time sent: 2014-02-18 20:03:09 PST (00:00:01 ago)
  Last time received: 2014-02-18 20:03:09 PST (00:00:01 ago)
  Maximum latency (secs): 0
  Minimum latency (secs): 0
```

Meaning In this example, the number of heartbeat request messages sent (**Heartbeats sent**) equals the number of heartbeat response messages received (**Heartbeats received**), with no heartbeat messages lost (**Heartbeats lost/missed**). This indicates that both member routers forming the heartbeat connection were available and operational. Any difference between **Heartbeats sent** and **Heartbeats received** appears in the **Heartbeats lost/missed** field.

The **Maximum latency** and **Minimum latency** fields measure the maximum and minimum number of seconds that elapse on the local router between trasnmission of a heartbeat request message and receipt of a heartbeat response message. In this example, the value **0** in the **Maximum latency** and **Minimum latency** fields indicates that there is no measurable network delay caused by this operation. You can use the **Maximum latency** value to determine whether you need to increase the **heartbeat-timeout** to a value higher than the default (2 seconds). If the maximum latency in your network is too high to accommodate a 2-second **heartbeat-timeout** value, increasing the **heartbeat-timeout** interval enables you to account for network delay when a Virtual Chassis adjacency disruption or split occurs.

Release History Table

Release	Description
14.1	Starting in Junos OS Release 14.1, you can configure a heartbeat connection in an MX Series Virtual Chassis.

Related Documentation

- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets on page 207
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- Virtual Chassis Heartbeat Connection Overview on page 41
- Global Roles and Local Roles in a Virtual Chassis on page 30
- *Configuring a Consistent Management IP Address*

Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets

A *heartbeat connection* is an IP-based, bidirectional packet connection in an MX Series Virtual Chassis between the Virtual Chassis master and backup routers. The *heartbeat packets* exchanged over this connection provide critical information about the availability and health of each member router. Starting in Junos OS Release 14.1, you can configure a heartbeat connection in an MX Series Virtual Chassis.

This example describes how to configure a heartbeat connection in an MX Series Virtual Chassis when the member routers reside in different subnets. For information about configuring a heartbeat connection when the Virtual Chassis member routers reside in the same subnet, see "Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet" on page 196.

- Requirements on page 207
- Overview on page 208
- Configuration on page 210
- Verification on page 219

Requirements

This example uses the following software and hardware components:

- Junos OS Release 14.1 and later releases
- One MX240 3D Universal Edge Router
- One MX480 3D Universal Edge Router

This configuration example has been tested using the software release listed and is assumed to work on all later releases.

BEST PRACTICE: We recommend that you use the **commit synchronize** command to save any configuration changes to the Virtual Chassis. For an MX Series Virtual Chassis, the **force** option is the default and only behavior when you issue the **commit synchronize** command. Issuing the **commit synchronize** command for an MX Series Virtual Chassis configuration has the same effect as issuing the **commit synchronize force** command.

Before you configure a heartbeat connection for a Virtual Chassis:

- Configure a Virtual Chassis consisting of two MX Series routers.

  See "Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis" on page 75

  As part of the preprovisioned Virtual Chassis configuration shown in the configuration example, you must create and apply the **member0-re0**, **member0-re1**, **member1-re0**, and **member1-re1** configuration groups for each member Routing Engine. Each configuration group includes a unique IP address for the management Ethernet interface (**fxp0**) on each Routing Engine.

  NOTE: When you create the preprovisioned Virtual Chassis configuration at the **[edit virtual-chassis]** hierarchy level, make sure you do *not* configure the **no-split-detection** statement to disable detection of a split in the Virtual Chassis. Using the **no-split-detection** statement is prohibited when you configure a Virtual Chassis heartbeat connection, and doing so causes the commit operation to fail.

- Ensure TCP connectivity between the master Routing Engine in the Virtual Chassis master router (VC-Mm) and the master Routing Engine in the Virtual Chassis backup router (VC-Bm).

  The Virtual Chassis heartbeat connection opens a proprietary TCP port numbered 33087 on the VC-Mm to listen for heartbeat messages. If your network design includes firewalls or filters, make sure the network allows traffic between TCP port 33087 on the VC-Mm and the dynamically allocated TCP port on the VC-Bm.

Overview

A *heartbeat connection* is an IP-based, bidirectional packet connection between the master router and backup router in an MX Series Virtual Chassis. The member routers forming the heartbeat connection exchange *heartbeat packets* that provide critical information about the availability and health of each member router. During a disruption or split in the Virtual Chassis configuration, the heartbeat connection prevents the member routers from changing mastership roles unnecessarily, which can cause undesirable results.

This example configures a heartbeat connection for an MX Series Virtual Chassis in which the two member routers, each with dual Routing Engines installed, reside in different

subnets. Member router **gladius** resides in subnet 10.4.0.0/16 and is the global master router for the Virtual Chassis (VC-M). Member router **trefoil** resides in subnet 10.5.0.0/16 and is the global backup router (VC-B) for the Virtual Chassis. The heartbeat connection is configured between the master Routing Engine in **gladius** (represented by VC-Mm or **member0-re0**) and the master Routing Engine in **trefoil** (represented by VC-Bm or **member1-re0**).

Configuring a heartbeat connection for an MX Series Virtual Chassis when the member routers reside in different subnets consists of the following tasks:

1. Configure two **master-only** IP addresses for the **fxp0** management interface: one for the member routers in subnet 10.4.0.0, and a diffferent address for the member routers in subnet 10.5.0.0.
2. Configure a network path for the heartbeat connection to ensure that both member routers can reach each other's networks.

   This example creates static routes to both subnet 10.4.0.0 and subnet 10.5.0.0 on each member router.
3. Configure the Virtual Chassis heartbeat address for each member Routing Engine to cross-connect to the **master-only** IP address for the corresponding member Routing Engine in the other subnet.
4. (Optional) Configure a nondefault value for the Virtual Chassis heartbeat timeout interval.

To establish the heartbeat connection in a two-member MX Series Virtual Chassis, you must configure the heartbeat address to establish the connection between the master and backup member routers. To ensure consistent access to the master Routing Engine in the Virtual Chassis master router (VC-Mm) regardless of which Routing Engine is currently active, you set the heartbeat address to the previously configured **master-only** IP address for the **fxp0** management interface.

Because the Virtual Chassis member routers in this example are in different subnets, you must configure a heartbeat address for each Routing Engine to enable a cross-connection to the **master-only** IP address for the corresponding Routing Engine in the other subnet, as shown in Table 24 on page 209:

Table 24: Heartbeat Cross-Connections for Member Routers in Different Subnets

Routing Engine	Subnet	Cross-connected Routing Engine	Heartbeat Address
member0-re0	10.4.0.0/16	**member1-re0**	10.5.2.210
member0-re1	10.4.0.0/16	**member1-re1**	10.5.2.210
member1-re0	10.5.0.0/16	**member0-re0**	10.4.2.210
member1-re1	10.5.0.0/16	**member0-re1**	10.4.2.210

Topology

This example configures a heartbeat connection for an MX Series Virtual Chassis with member routers residing in different subnets. For redundancy, each member router is configured with two Virtual Chassis ports.

Table 25 on page 210 shows the hardware and software configuration settings for each MX Series router in the Virtual Chassis.

Table 25: Components of the Sample MX Series Virtual Chassis with Member Routers in Different Subnets

Router Name	Hardware	Serial Number	Member ID	Role	Virtual Chassis Ports	Subnet
gladius	MX240 router with: • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member0-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member0-re1**)	JN10C7135AFC	0	routing-engine (master)	vcp-2/2/0 vcp-2/3/0	10.4.0.0/16
trefoil	MX480 router with: • Master RE-S-2000 Routing Engine in slot 0 (represented in example as **member1-re0**) • Backup RE-S-2000 Routing Engine in slot 1 (represented in example as **member1-re1**)	JN115D117AFB	1	routing-engine (backup)	vcp-2/0/0 vcp-5/2/0	10.5.0.0/16

Configuration

To configure a heartbeat connection in an MX Series Virtual Chassis with member routers in different subnets, perform these tasks:

- Configuring a Consistent Management IP Address for Each Routing Engine on page 212
- Configuring Static Routes for Both Subnets on Each Routing Engine on page 214
- Configuring the Heartbeat Address and Heartbeat Timeout on page 217

CLI Quick Configuration

To quickly configure a heartbeat connection for a Virtual Chassis with with member routers in different subnets, copy the following commands and paste them into the router terminal window:

```
[edit]
set groups member0-re0 interfaces fxp0 unit 0 family inet address 10.4.2.210/16
    master-only
set groups member0-re1 interfaces fxp0 unit 0 family inet address 10.4.2.210/16
    master-only
set groups member1-re0 interfaces fxp0 unit 0 family inet address 10.5.2.210/16
    master-only
set groups member1-re1 interfaces fxp0 unit 0 family inet address 10.5.2.210/16
    master-only
set groups member0-re0 routing-options static route 10.4.0.0/16 next-hop 10.4.0.1
set groups member0-re0 routing-options static route 10.4.0.0/16 retain
set groups member0-re0 routing-options static route 10.4.0.0/16 no-readvertise
set groups member0-re0 routing-options static route 10.5.0.0/16 next-hop 10.4.0.1
set groups member0-re0 routing-options static route 10.5.0.0/16 retain
set groups member0-re0 routing-options static route 10.5.0.0/16 no-readvertise
set groups member0-re1 routing-options static route 10.4.0.0/16 next-hop 10.4.0.1
set groups member0-re1 routing-options static route 10.4.0.0/16 retain
set groups member0-re1 routing-options static route 10.4.0.0/16 no-readvertise
set groups member0-re1 routing-options static route 10.5.0.0/16 next-hop 10.4.0.1
set groups member0-re1 routing-options static route 10.5.0.0/16 retain
set groups member0-re1 routing-options static route 10.5.0.0/16 no-readvertise
set groups member1-re0 routing-options static route 10.5.0.0/16 next-hop 10.5.0.1
set groups member1-re0 routing-options static route 10.5.0.0/16 retain
set groups member1-re0 routing-options static route 10.5.0.0/16 no-readvertise
set groups member1-re0 routing-options static route 10.4.0.0/16 next-hop 10.5.0.1
set groups member1-re0 routing-options static route 10.4.0.0/16 retain
set groups member1-re0 routing-options static route 10.4.0.0/16 no-readvertise
set groups member1-re1 routing-options static route 10.5.0.0/16 next-hop 10.5.0.1
set groups member1-re1 routing-options static route 10.5.0.0/16 retain
set groups member1-re1 routing-options static route 10.5.0.0/16 no-readvertise
set groups member1-re1 routing-options static route 10.4.0.0/16 next-hop 10.5.0.1
set groups member1-re1 routing-options static route 10.4.0.0/16 retain
set groups member1-re1 routing-options static route 10.4.0.0/16 no-readvertise
set groups member0-re0 virtual-chassis heartbeat-address 10.5.2.210
set groups member0-re1 virtual-chassis heartbeat-address 10.5.2.210
set groups member1-re0 virtual-chassis heartbeat-address 10.4.2.210
set groups member1-re1 virtual-chassis heartbeat-address 10.4.2.210
set virtual-chassis heartbeat-timeout 10
```

Configuring a Consistent Management IP Address for Each Routing Engine

Step-by-Step Procedure

In addition to configuring a unique IP address for the **fxp0** management interface on each Routing Engine when you first set up the Virtual Chassis, you must configure additional management IP addresses, known as the **master-only** address, to ensure consistent access to the **fxp0** management interface on the master Routing Engine in the Virtual Chassis master router (VC-Mm). The **master-only** address is active only on the management interface for the VC-Mm. During a switchover, the **master-only** address moves to the new Routing Engine currently functioning as the VC-Mm.

Because the Virtual Chassis master router and backup router in this example reside in different subnets, you must configure two different **master-only** IP addresses: one for the Routing Engines in subnet 10.4.0.0/16 (**member0-re0** and **member0-re1**), and one for the Routing Engines in subnet 10.5.0.0/16 (**member1-re0** and **member1-re1**). You then configure these **master-only** addresses as the subnet-specific heartbeat addresses to establish the heartbeat connection. For more information about the cross-connections in this example, see Table 24 on page 209.

To configure the master-only **fxp0** IP address for each Routing Engine:

1. From the console on member 0, configure the IP address for the **fxp0** management interface for the Routing Engines in subnet 10.4.0.0/16.

```
{master:member0-re0}[edit]
user@gladius# set groups member0-re0 interfaces fxp0 unit 0 family inet address
  10.4.2.210/16 master-only
user@gladius# set groups member0-re1 interfaces fxp0 unit 0 family inet address
  10.4.2.210/16 master-only
```

2. From the console on member 0, configure the IP address for the **fxp0** management interface for the Routing Engines in subnet 10.5.0.0/16.

```
{master:member0-re0}[edit]
user@gladius# set groups member1-re0 interfaces fxp0 unit 0 family inet address
  10.5.2.210/16 master-only
user@gladius# set groups member1-re1 interfaces fxp0 unit 0 family inet address
  10.5.2.210/16 master-only
```

Results

From the console on the Virtual Chassis master router, display the results of the configuration. For brevity, portions of the configuration unrelated to this procedure are replaced by an ellipsis (...).

For **member0-re0**:

```
{master:member0-re0}[edit]
user@gladius# show groups member0-re0
system {
   host-name gladius;
   backup-router 10.4.0.1 destination [ 172.16.0.0/12 ... 10.204.0.0/16 ];
}
interfaces {
```

```
    fxp0 {
      unit 0 {
        family inet {
          address 10.4.2.100/16;
          address 10.4.2.210/16 {
            master-only;
          }
        }
      }
    }
  }
```

For **member0-re1**:

```
  {master:member0-re0}[edit]
  user@gladius# show groups member0-re1
  system {
    host-name gladius1;
    backup-router 10.4.0.1 destination [ 172.16.0.0/12 ... 10.204.0.0/16 ];
  }
  interfaces {
    fxp0 {
      unit 0 {
        family inet {
          address 10.4.2.101/16;
          address 10.4.2.210/16 {
            master-only;
          }
        }
      }
    }
  }
```

For **member1-re0**:

```
  {master:member0-re0}[edit]
  user@gladius# show groups member1-re0
  system {
    host-name trefoil;
    backup-router 10.5.0.1 destination [ 172.16.0.0/12 ... 10.204.0.0/16 ];
  }
  interfaces {
    fxp0 {
      unit 0 {
        family inet {
          address 10.5.3.101/16;
          address 10.5.2.210/16 {
            master-only;
          }
        }
      }
    }
  }
```

For **member1-re1**:

```
  {master:member0-re0}[edit]
```

```
user@gladius# show groups member1-re1
system {
   host-name trefoil1;
   backup-router 10.5.0.1 destination [ 172.16.0.0/12 ... 10.204.0.0/16 ];
}
interfaces {
   fxp0 {
      unit 0 {
         family inet {
            address 10.5.3.102/16;
            address 10.5.2.210/16 {
               master-only;
            }
         }
      }
   }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring Static Routes for Both Subnets on Each Routing Engine

Step-by-Step Procedure

You must configure secure and reliable routes for subnets 10.4.0.0/16 and 10.5.0.0/16 on each Routing Engine for the exchange of TCP/IP heartbeat packets. The heartbeat packets provide critical information about the availability and health of each member router.

The routes you configure for the heartbeat connection must be independent of the Virtual Chassis port links. Specifically, you must ensure that the master Routing Engine in the Virtual Chassis backup router (VC-Bm) can make a TCP/IP connection to the **master-only** IP address of the master Routing Engine in the Virtual Chassis master router (VC-Mm).

This examples creates static routes to both subnets on each member Routing Engine to configure the heartbeat path. However, you can choose the method that best meets your needs to configure the heartbeat path for member routers in different subnets.

BEST PRACTICE: We recommend that you use the router management interface (**fxp0**) as the heartbeat path. The management interface is generally available earlier than the line card interfaces, and is typically connected to a more secure network than the other interfaces.

To create static routes for subnets 10.4.0.0/16 and 10.5.0.0/16 on each Routing Engine:

1. Log in to the console on member 0 (Virtual Chassis master router).

2. Configure the static routes for **member0-re0**.

```
{master:member0-re0}[edit]
user@gladius# set groups member0-re0 routing-options static route 10.4.0.0/16
  next-hop 10.4.0.1
```

```
user@gladius# set groups member0-re0 routing-options static route 10.4.0.0/16
  retain
user@gladius# set groups member0-re0 routing-options static route 10.4.0.0/16
  no-readvertise
user@gladius# set groups member0-re0 routing-options static route 10.5.0.0/16
  next-hop 10.4.0.1
user@gladius# set groups member0-re0 routing-options static route 10.5.0.0/16
  retain
user@gladius# set groups member0-re0 routing-options static route 10.5.0.0/16
  no-readvertise
```

3. Configure the static routes for **member0-re1**.

```
{master:member0-re0}[edit]
user@gladius# set groups member0-re1 routing-options static route 10.4.0.0/16
  next-hop 10.4.0.1
user@gladius# set groups member0-re1 routing-options static route 10.4.0.0/16
  retain
user@gladius# set groups member0-re1 routing-options static route 10.4.0.0/16
  no-readvertise
user@gladius# set groups member0-re1 routing-options static route 10.5.0.0/16
  next-hop 10.4.0.1
user@gladius# set groups member0-re1 routing-options static route 10.5.0.0/16
  retain
user@gladius# set groups member0-re1 routing-options static route 10.5.0.0/16
  no-readvertise
```

4. Configure the static routes for **member1-re0**.

```
{master:member0-re0}[edit]
user@gladius# set groups member1-re0 routing-options static route 10.5.0.0/16
  next-hop 10.5.0.1
user@gladius# set groups member1-re0 routing-options static route 10.5.0.0/16
  retain
user@gladius# set groups member1-re0 routing-options static route 10.5.0.0/16
  no-readvertise
user@gladius# set groups member1-re0 routing-options static route 10.4.0.0/16
  next-hop 10.5.0.1
user@gladius# set groups member1-re0 routing-options static route 10.4.0.0/16
  retain
user@gladius# set groups member1-re0 routing-options static route 10.4.0.0/16
  no-readvertise
```

5. Configure the static routes for **member1-re1**.

```
{master:member0-re0}[edit]
user@gladius# set groups member1-re1 routing-options static route 10.5.0.0/16
  next-hop 10.5.0.1
user@gladius# set groups member1-re1 routing-options static route 10.5.0.0/16
  retain
user@gladius# set groups member1-re1 routing-options static route 10.5.0.0/16
  no-readvertise
user@gladius# set groups member1-re1 routing-options static route 10.4.0.0/16
  next-hop 10.5.0.1
```

```
user@gladius# set groups member1-re1 routing-options static route 10.4.0.0/16
  retain
user@gladius# set groups member1-re1 routing-options static route 10.4.0.0/16
  no-readvertise
```

Results Display the results of the configuration. For brevity, portions of the configuration unrelated to this procedure are replaced by an ellipsis (...).

For **member0-re0**:

```
{master:member0-re0}[edit]
user@gladius# show groups member0-re0 routing-options static
route 10.4.0.0/16 {
  next-hop 10.4.0.1;
  retain;
  no-readvertise;
}
route 10.5.0.0/16 {
  next-hop 10.4.0.1;
  retain;
  no-readvertise;
}
...
```

For **member0-re1**:

```
{master:member0-re0}[edit]
user@gladius# show groups member0-re1 routing-options static
route 10.4.0.0/16 {
  next-hop 10.4.0.1;
  retain;
  no-readvertise;
}
route 10.5.0.0/16 {
  next-hop 10.4.0.1;
  retain;
  no-readvertise;
}
...
```

For **member1-re0**:

```
{master:member0-re0}[edit]
user@gladius# show groups member1-re0 routing-options static
route 10.5.0.0/16 {
  next-hop 10.5.0.1;
  retain;
  no-readvertise;
}
route 10.4.0.0/16 {
  next-hop 10.5.0.1;
  retain;
  no-readvertise;
}
...
```


For **member1-re1**:

```
{master:member0-re0}[edit]
user@gladius# show groups member1-re1 routing-options static
route 10.5.0.0/16 {
    next-hop 10.5.0.1;
    retain;
    no-readvertise;
}
route 10.4.0.0/16 {
    next-hop 10.5.0.1;
    retain;
    no-readvertise;
}
...
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring the Heartbeat Address and Heartbeat Timeout

Step-by-Step Procedure

To enable cross-connection between Virtual Chassis member routers in different subnets, you configure 10.5.2.210, which is the **master-only** IP address for the Routing Engines in subnet 10.5.0.0/16, as the heartbeat address for the Routing Engines in subnet 10.4.0.0/16 (**member0-re0** and **member0-re1**). Conversely, you configure 10.4.2.210, which is the **master-only** IP address for the Routing Engines in subnet 10.4.0.0/16, as the heartbeat address for the Routing Engines in subnet 10.5.0.0/16 (**member1-re0** and **member1-re1**). For more information about the cross-connections in this example, see Table 24 on page 209.

Optionally, you can also configure a nondefault value for the heartbeat timeout interval. The heartbeat timeout is the maximum time within which a Virtual Chassis member router must respond to a heartbeat packet sent by the other member router. If you do not explicitly configure the heartbeat timeout interval, the default value (2 seconds) applies.

To configure the heartbeat address and heartbeat timeout:

1. Log in to the console on member 0 (Virtual Chassis master router).

2. Configure the heartbeat address for each Routing Engine.

```
{master:member0-re0}[edit]
user@gladius# set groups member0-re0 virtual-chassis heartbeat-address 10.5.2.210
user@gladius# set groups member0-re1 virtual-chassis heartbeat-address 10.5.2.210
user@gladius# set groups member1-re0 virtual-chassis heartbeat-address 10.4.2.210
user@gladius# set groups member1-re1 virtual-chassis heartbeat-address 10.4.2.210
```

3. (Optional) Configure a nondefault value for the heartbeat timeout interval.

```
{master:member0-re0}[edit]
user@gladius# set virtual-chassis heartbeat-timeout 10
```

Results Display the results of the configuration.

For **member0-re0**:

```
{master:member0-re0}[edit]
user@gladius# show groups member0-re0 virtual-chassis
preprovisioned;
traceoptions {
    file VCCP size 100m;
    flag all;
}
heartbeat-address 10.5.2.210;
heartbeat-timeout 10;
member 0 {
    role routing-engine;
    serial-number JN10C7135AFC;
}
member 1 {
    role routing-engine;
    serial-number JN115D117AFB;
}
```

For **member0-re1**:

```
{master:member0-re0}[edit]
user@gladius# show groups member0-re1 virtual-chassis
preprovisioned;
traceoptions {
    file VCCP size 100m;
    flag all;
}
heartbeat-address 10.5.2.210;
heartbeat-timeout 10;
member 0 {
    role routing-engine;
    serial-number JN10C7135AFC;
}
member 1 {
    role routing-engine;
    serial-number JN115D117AFB;
}
```

For **member1-re0**:

```
{master:member0-re0}[edit]
user@gladius# show groups member1-re0 virtual-chassis
preprovisioned;
traceoptions {
    file VCCP size 100m;
    flag all;
}
heartbeat-address 10.4.2.210;
heartbeat-timeout 10;
member 0 {
    role routing-engine;
    serial-number JN10C7135AFC;
}
```



```
  member 1 {
    role routing-engine;
    serial-number JN115D117AFB;
  }
```

For **member1-re1**:

```
{master:member0-re0}[edit]
user@gladius# show groups member1-re1 virtual-chassis
preprovisioned;
traceoptions {
    file VCCP size 100m;
    flag all;
}
heartbeat-address 10.4.2.210;
heartbeat-timeout 10;
member 0 {
    role routing-engine;
    serial-number JN10C7135AFC;
}
member 1 {
    role routing-engine;
    serial-number JN115D117AFB;
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

To confirm that the Virtual Chassis heartbeat connection is working properly, perform these tasks:

- Verifying the Virtual Chassis Heartbeat Connection on page 219
- Verifying Use of the Heartbeat Connection During an Adjacency Split or Disruption on page 220
- Verifying Virtual Chassis Member Health from Heartbeat Statistics on page 221

Verifying the Virtual Chassis Heartbeat Connection

Purpose Verify that the heartbeat connection between the Virtual Chassis member routers is properly configured and operational.

Action Display the state of one or both member routers when a heartbeat connection is configured.

```
{master:member0-re0}

user@gladius> show virtual-chassis heartbeat
member0:
--------------------------------------------------------------------------
Local        Remote          State        Time
10.4.2.210   10.5.3.101      Alive        2014-03-18 10:18:14 PST

member1:
--------------------------------------------------------------------------
Local        Remote          State        Time
10.5.3.101   10.4.2.210      Alive        2014-03-18 10:18:15 PST
```

Meaning For each member router, the command output displays the IP addresses of the local and remote member routers that form the heartbeat connection. The value **Alive** in the **State** field confirms that the master Routing Engine in the specified member router is connected and has received a heartbeat response message. The **Time** field specifies the date and time of the last connection state change.

Verifying Use of the Heartbeat Connection During an Adjacency Split or Disruption

Purpose Verify use of the heartbeat connection when an adjacency disruption or split is detected in the Virtual Chassis.

Action Display the status of the member routers in the Virtual Chassis:

```
{master:member0-re0}

user@gladius> show virtual-chassis status
Preprovisioned Virtual Chassis
Virtual Chassis ID: a5b6.be0c.9525
                                                 Mastership           Neighbor List
Member ID       Status   Serial No    Model    priority    Role      ID  Interface

0 (FPC   0- 11) Heartbt  JN10C7135AFC mx240          129  Master*    1  vcp-2/2/0
                                                                     1  vcp-2/3/0

1 (FPC  12- 23) Prsnt    JN115D117AFB mx480          129  Backup     0  vcp-2/0/0
                                                                     0  vcp-5/2/0
```

Meaning The **Status** field for member ID 0 displays **Heartbt**, which indicates that this member router has used the heartbeat packet connection to maintain mastership roles during an adjacency disruption or split in the Virtual Chassis configuration. The **Status** field for member ID 1 displays **Prsnt**, which indicates that this member router is connected to the Virtual Chassis.

If a router is not currently connected to the Virtual Chassis, the **Status** field displays **NotPrsnt**.

Verifying Virtual Chassis Member Health from Heartbeat Statistics

Purpose Use statistics collected by the heartbeat connection to verify the availability and health of each Virtual Chassis member router. You can also use the **show virtual-chassis heartbeat detail** command to determine the maximum latency and minimum latency in your network.

Action Display and review the statistics collected by the heartbeat connection.

```
{master:member0-re0}

user@gladius> show virtual-chassis heartbeat detail
member0:
--------------------------------------------------------------------------
Local           Remote          State        Time
10.4.2.210      10.5.3.101      Alive        2014-03-18 10:18:14 PST

Heartbeat statistics
  Heartbeats sent: 10079
  Heartbeats received: 10079
  Heartbeats lost/missed: 0
  Last time sent: 2014-03-18 20:03:10 PST (00:00:00 ago)
  Last time received: 2014-03-18 20:03:10 PST (00:00:00 ago)
  Maximum latency (secs): 0
  Minimum latency (secs): 0

member1:
--------------------------------------------------------------------------
Local           Remote          State        Time
10.5.3.101      10.4.2.210      Alive        2014-03-18 10:18:15 PST

Heartbeat statistics
  Heartbeats sent: 10083
  Heartbeats received: 10083
  Heartbeats lost/missed: 0
  Last time sent: 2014-02-18 20:03:09 PST (00:00:01 ago)
  Last time received: 2014-02-18 20:03:09 PST (00:00:01 ago)
  Maximum latency (secs): 0
  Minimum latency (secs): 0
```

Meaning In this example, the number of heartbeat request messages sent (**Heartbeats sent**) equals the number of heartbeat response messages received (**Heartbeats received**), with no heartbeat messages lost (**Heartbeats lost/missed**). This indicates that both member routers forming the heartbeat connection are available and operational. Any difference between **Heartbeats sent** and **Heartbeats received** appears in the **Heartbeats lost/missed** field.

The **Maximum latency** and **Minimum latency** fields measure the maximum and minimum number of seconds that elapse on the local router between transmission of a heartbeat request message and receipt of a heartbeat response message. In this example, the value **0** in the **Maximum latency** and **Minimum latency** fields indicates that there is no measurable

network delay caused by this operation. You can use the **Maximum latency** value to determine whether you need to increase the **heartbeat-timeout** to a value higher than the default (2 seconds). If the maximum latency in your network is too high to accommodate a 2-second **heartbeat-timeout** value, increasing the **heartbeat-timeout** interval enables you to account for network delay when a Virtual Chassis adjacency disruption or split occurs.

Release History Table

Release	Description
14.1	Starting in Junos OS Release 14.1, you can configure a heartbeat connection in an MX Series Virtual Chassis.

Related Documentation

- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet on page 196
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- Virtual Chassis Heartbeat Connection Overview on page 41
- Global Roles and Local Roles in a Virtual Chassis on page 30
- *Configuring a Consistent Management IP Address*

CHAPTER 11

Tracing Virtual Chassis Operations for Troubleshooting Purposes

- Tracing Virtual Chassis Operations for MX Series 3D Universal Edge Routers on page 224
- Configuring the Name of the Virtual Chassis Trace Log File on page 225
- Configuring Characteristics of the Virtual Chassis Trace Log File on page 225
- Configuring Access to the Virtual Chassis Trace Log File on page 226
- Using Regular Expressions to Refine the Output of the Virtual Chassis Trace Log File on page 227
- Configuring the Virtual Chassis Operations to Trace on page 228

Tracing Virtual Chassis Operations for MX Series 3D Universal Edge Routers

The Junos OS trace feature tracks Virtual Chassis operations and records events in a log file. The error descriptions captured in the log file provide detailed information to help you solve problems.

By default, tracing is disabled. When you enable the tracing operation on the router to be configured as the master (also referred to as the *protocol master*) of an MX Series Virtual Chassis, the default tracing behavior is as follows:

1. Important events are logged in a file with the name you specify in the **/var/log** directory. You cannot change the directory (**/var/log**) in which trace files are located.
2. When a trace file named ***trace-file*** reaches its maximum size, it is renamed ***trace-file*.0**, then ***trace-file*.1**, and so on, until the maximum number of trace files is reached. Then the oldest trace file is overwritten.

   You can optionally specify the maximum number of trace files to be from 2 through 1000. You can also configure the maximum file size to be from 10 KB through 1 gigabyte (GB). (For more information about how log files are created, see the System Log Explorer.)

   By default, only the user who configures the tracing operation can access log files. You can optionally configure read-only access for all users.

To configure tracing of MX Series Virtual Chassis operations:

1. Configure a filename for the trace log.

   See "Configuring the Name of the Virtual Chassis Trace Log File" on page 225.

2. (Optional) Configure characteristics of the trace log file.

   See "Configuring Characteristics of the Virtual Chassis Trace Log File" on page 225.

3. (Optional) Configure user access to the trace log file.

   See "Configuring Access to the Virtual Chassis Trace Log File" on page 226.

4. (Optional) Refine the output of the trace log file.

   See "Using Regular Expressions to Refine the Output of the Virtual Chassis Trace Log File" on page 227.

5. Configure flags to specify the Virtual Chassis operations that you want to trace.

   See "Configuring the Virtual Chassis Operations to Trace" on page 228.

Related Documentation

- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66

- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Configuring the Name of the Virtual Chassis Trace Log File

To trace operations for a Virtual Chassis, you must configure the name of the trace log file that the software saves in the **/var/log** directory.

To configure the filename for tracing Virtual Chassis operations:

- On the device to be designated as the master of the Virtual Chassis, specify the name of the trace log file.

```
[edit virtual-chassis]
user@host# set traceoptions file filename
```

Related Documentation

- Tracing Virtual Chassis Operations for MX Series 3D Universal Edge Routers on page 224
- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Configuring Characteristics of the Virtual Chassis Trace Log File

You can optionally configure the following characteristics of the trace log file for a Virtual Chassis:

- Maximum number of trace files—When a trace file named ***trace-file*** reaches its maximum size, it is renamed ***trace-file*.0**, then ***trace-file*.1**, and so on, until the maximum number of trace files is reached. Then the oldest trace file is overwritten. You can optionally specify the maximum number of trace files to be from 2 through 1000. If you specify a maximum number of files with the **files** option, you must also specify a maximum file size with the **size** option.
- Maximum trace file size—You can configure the maximum trace file size to be from 10 KB through 1 gigabyte (GB) If you specify a maximum file size with the **size** option, you must also specify a maximum number of files with the **files** option.
- Timestamp—By default, timestamp information is placed at the beginning of each line of trace output. You can optionally prevent placement of a timestamp on any trace log file.
- Appending or replacing the trace file—By default, the router or switch appends new information to an existing trace file. You can optionally specify that the router or switch replace an exisiting trace file instead of appending information to it.

To configure the maximum number and maximum size of trace files:

- On the router or switch to be designated as the master of the Virtual Chassis, specify the maximum number and maximum size of the trace file.

```
[edit virtual-chassis]
user@host# set traceoptions file filename files number size maximum-file-size
```

For example, to set the maximum number of files to 20 and the maximum file size to 2 MB for a trace file named **vccp**:

```
[edit virtual-chassis]
user@host# set traceoptions file vccp files 20 size 2097152
```

When the **vccp** trace file for this example reaches 2 MB, **vccp** is renamed **vccp.0**, and a new file named **vccp** is created. When the new **vccp** file reaches 2 MB, **vccp.0** is renamed **vccp.1** and **vccp** is renamed **vccp.0**. This process repeats until there are 20 trace files. Then the oldest file (**vccp.19**) is overwritten by the newest file (**vccp.0**).

To prevent the router or switch from the placing a timestamp on the trace log file:

- On the router or switch to be designated as the master of the Virtual Chassis, specify that a timestamp not appear on the trace log file:

```
[edit virtual-chassis]
user@host# set traceoptions file filename no-stamp
```

To replace an exisiting trace file instead of appending information to it:

- On the router or switch to be designated as the master of the Virtual Chassis, specify that the router or switch replaces an existing trace file:

```
[edit virtual-chassis]
user@host# set traceoptions file filename replace
```

Related Documentation

- Tracing Virtual Chassis Operations for MX Series 3D Universal Edge Routers on page 224
- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Configuring Access to the Virtual Chassis Trace Log File

By default, only the user who configures the tracing operation can access the log files. You can enable all users to read the log file, and you can explicitly set the default behavior of the log file.

To configure access to the trace log file for all users:

- On the router or switch to be designated as the master of the Virtual Chassis, specify that all users can read the trace log file.

```
[edit virtual-chassis]
user@host# set traceoptions file filename world-readable
```

To explicitly set the default behavior to enable access to the trace log file only for the user who configured tracing:

- On the router or switch to be designated as the master of the Virtual Chassis, specify that only the user who configured tracing can read the trace log file.

```
[edit virtual-chassis]
user@host# set traceoptions file filename no-world-readable
```

Related Documentation

- Tracing Virtual Chassis Operations for MX Series 3D Universal Edge Routers on page 224
- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Using Regular Expressions to Refine the Output of the Virtual Chassis Trace Log File

By default, the trace operation output includes all lines relevant to the logged events. You can refine the output of the trace log file for a Virtual Chassis by including regular expressions to be matched.

To refine the output of the trace log file:

- On the router or switch to be designated as the master of the Virtual Chassis, configure a regular expression to be matched.

```
[edit virtual-chassis]
user@host# set traceoptions file filename match regular-expression
```

Related Documentation

- Tracing Virtual Chassis Operations for MX Series 3D Universal Edge Routers on page 224
- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

Configuring the Virtual Chassis Operations to Trace

By default, the router or switch logs only important events. You can specify which operations to trace for a Virtual Chassis by including specific tracing flags when you configure tracing. Table 26 on page 228 describes the flags that you can include.

Table 26: Tracing Flags for Virtual Chassis

Flag	Description
all	Trace all operations.
auto-configuration	Trace Virtual Chassis ports that have been automatically configured.
csn	Trace Virtual Chassis complete sequence number (CSN) packets.
error	Trace Virtual Chassis errored packets.
graceful-restart	Trace Virtual Chassis graceful restart events.
hello	Trace Virtual Chassis hello packets.
krt	Trace Virtual Chassis kernel routing table (KRT) events.
lsp	Trace Virtual Chassis link-state packets.
lsp-generation	Trace Virtual Chassis link-state packet generation.
me	Trace Virtual Chassis mastership election (ME) events.
normal	Trace normal events.
packets	Trace Virtual Chassis packets.
parse	Trace reading of the configuration.
psn	Trace partial sequence number (PSN) packets.
route	Trace Virtual Chassis routing information.
spf	Trace Virtual Chassis shortest-path-first (SPF) events.
state	Trace Virtual Chassis state transitions.
task	Trace Virtual Chassis task operations.

To configure the flags for the Virtual Chassis operations to be logged:

1. Specify the tracing flag that represents the operation you want to trace.

   ```
   [edit virtual-chassis]
   user@host# set traceoptions flag flag
   ```

2. (Optional) Specify one or more of the following additional tracing options for the specified flag:

   - To generate detailed trace output, use the **detail** option.
   - To disable a particular flag, use the **disable** option.
   - To trace received packets, use the **receive** option.
   - To trace transmitted packets, use the **send** option.

   For example, to generate detailed trace output for Virtual Chassis mastership election events in received packets:

   ```
   [edit virtual-chassis]
   user@host# set traceoptions flag me detail receive
   ```

Related Documentation

- Tracing Virtual Chassis Operations for MX Series 3D Universal Edge Routers on page 224
- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

CHAPTER 12

Configuration Statements

- aggregated-ether-options on page 232
- heartbeat-address (MX Series Virtual Chassis) on page 234
- heartbeat-timeout (MX Series Virtual Chassis) on page 235
- heartbeat-tos (MX Series Virtual Chassis) on page 236
- locality-bias (MX Series Virtual Chassis) on page 237
- logical-interface-chassis-redundancy (MX Series Virtual Chassis) on page 238
- logical-interface-fpc-redundancy (Aggregated Ethernet Subscriber Interfaces) on page 238
- member (MX Series Virtual Chassis) on page 239
- network-services on page 240
- no-split-detection (MX Series Virtual Chassis) on page 241
- preprovisioned (MX Series Virtual Chassis) on page 242
- role (MX Series Virtual Chassis) on page 243
- sampling-instance on page 244
- serial-number (MX Series Virtual Chassis) on page 245
- targeted-distribution (Static Interfaces over Aggregated Ethernet) on page 246
- traceoptions (MX Series Virtual Chassis) on page 247
- virtual-chassis (MX Series Virtual Chassis) on page 249

aggregated-ether-options

Syntax

```
aggregated-ether-options {
  ethernet-switch-profile {
    ethernet-policer-profile {
      input-priority-map {
        ieee802.1p premium [ values ];
      }
      output-priority-map {
        classifier {
          premium {
            forwarding-class class-name {
              loss-priority (high | low);
            }
          }
        }
      }
      policer cos-policer-name {
        aggregate {
          bandwidth-limit bps;
          burst-size-limit bytes;
        }
        premium {
          bandwidth-limit bps;
          burst-size-limit bytes;
        }
      }
    }
    (mac-learn-enable | no-mac-learn-enable);
  }
  (flow-control | no-flow-control);
  lacp {
    (active | passive);
    link-protection {
      disable;
    (revertive | non-revertive);
    periodic interval;
    system-priority priority;
    system-id system-id;
  }
  link-protection;
  load-balance;
  link-speed speed;
  logical-interface-chassis-redundancy;
  logical-interface-fpc-redundancy;
  (loopback | no-loopback);
  minimum-links number;
  rebalance-periodic time hour:minute <interval hours>;
  source-address-filter {
    mac-address;
    (source-filtering | no-source-filtering);
  }
}
```

Hierarchy Level [edit interfaces ae*x*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Configure aggregated Ethernet-specific interface properties.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level
interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- *Ethernet Interfaces Overview*

heartbeat-address (MX Series Virtual Chassis)

Syntax heartbeat-address (*ip-address* | *ipv6-address*);

Hierarchy Level [edit virtual-chassis]

Release Information Statement introduced in Junos OS Release 14.1.

Description Configure an IPv4 address or IPv6 address used to create an IP-based packet connection, known as a *heartbeat connection*, between the master router and backup router in an MX Series Virtual Chassis. To ensure consistent access to the master Routing Engine in the Virtual Chassis master router (VC-Mm) regardless of which Routing Engine is active, you must configure the address for the management Ethernet interface (**fxp0**) with the **master-only** statement at the **[edit groups]** hierarchy level.

NOTE: The **heartbeat-address** statement and the **no-split-detection** statement at the **[edit virtual-chassis]** hierarchy level are mutually exclusive. As a result, the software prevents you from configuring both **heartbeat-address** and **no-split-detection** at the same time. If you attempt to do so, the software displays an error message and causes the **commit** operation to fail.

Options ***ip-address***—IPv4 **master-only** address for the **fxp0** management interface.

ipv6-address—IPv6 **master-only** address for the **fxp0** management interface.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet on page 196
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets on page 207

heartbeat-timeout (MX Series Virtual Chassis)

Syntax heartbeat-timeout *seconds*;

Hierarchy Level [edit virtual-chassis]

Release Information Statement introduced in Junos OS Release 14.1.

Description Configure the maximum time period within which a Virtual Chassis member router must respond to a heartbeat packet sent by the other member router. When an adjacency disruption or split is detected in the Virtual Chassis configuration, each member router sends a single heartbeat packet to the other member router to determine whether that router is still operating and able to respond. If the other member router responds to the heartbeat packet within the configured or default timeout period, the heartbeat connection helps prevent the member routers from changing mastership roles, which can cause undesirable results.

NOTE: Both member routers forming the heartbeat connection must be assigned the routing-engine role in the preprovisioned Virtual Chassis configuration. The routing-engine role makes the router eligible to function as either the master router or backup router of the Virtual Chassis.

Options *seconds*—Number of seconds within which a Virtual Chassis member router must respond to a heartbeat packet.

Range: 1 through 60 seconds

Default: 2 seconds

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet on page 196
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets on page 207

heartbeat-tos (MX Series Virtual Chassis)

Syntax heartbeat-tos *value*;

Hierarchy Level [edit virtual-chassis]

Release Information Statement introduced in Junos OS Release 15.1.

Description Configure a network-specific value for the header field that defines either the type of service (ToS), traffic class (TC), or differential services (DS) octet that is transmitted in the header of a heartbeat packet.. This field is a single octet that ensures preferential delivery of heartbeat packets between Virtual Chassis member routers in a congested network. You can set the IP precedence portion of the ToS header field or the differentiated services code point (DSCP) in the DS header field in accordance with your network. When you configure a **heartbeat-tos** value, the software uses this octet value in the IP header for all generated Virtual Chassis heartbeat packets.

Configuring the **heartbeat-tos** Virtual Chassis heartbeat packets is particularly useful in congested networks or in networks with high bandwidth demands. For example, setting the appropriate **heartbeat-tos** value for your network design enables you to prioritize the traffic flows that transmit Virtual Chassis heartbeat packets at a higher service level than the traffic flows for traditional IP applications such as e-mail or Web browsing. During periods of network congestion, configuring the appropriate value can ensure reliable delivery of the heartbeat packets and decrease the likelihood of packet loss.

BEST PRACTICE: **To properly prioritize the traffic flows for heartbeat packets based on the network load, make sure you are using a value that is consistent with the protocol standard that was implemented in your network.**

Options ***value***—Network-specific value of the ToS, TC, or DS octet transmitted in Virtual Chassis heartbeat packets. Depending on your network design and protocol family implementation, the value you specify must conform to the following RFC standards:

- IPv4 ToS as defined in RFC 791, *INTERNET PROTOCOL - DARPA INTERNET PROGRAM PROTOCOL SPECIFICATION*
- IPv6 traffic class (TC) as defined in RFC 2460, *Internet Protocol, Version 6 (IPv6) Specification*
- IPv4 and IPv6 Differentiated Services as defined in RFC 2474, *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
- Assured forwarding per-hop behavior (PHB) as defined in RFC 2597, *Assured Forwarding PHB Group*
- RFC 3260, *New Terminology and Clarifications for Diffserv*

Range: 0 through 255

Default: 0 (**heartbeat-tos** not set)

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet on page 196
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets on page 207
- Virtual Chassis Heartbeat Connection Overview on page 41

locality-bias (MX Series Virtual Chassis)

Syntax
```
locality-bias;
```

Hierarchy Level
[edit virtual-chassis]

Release Information
Statement introduced in Junos OS Release 14.1.

Description
Configure unicast transit traffic for equal-cost multipath (ECMP) groups and aggregated Ethernet bundles to egress links in the same (local) member router in an MX Series Virtual Chassis rather than to egress links in the remote member router, provided that the local member router has an equal or larger number of available egress links than the remote member router.

Required Privilege Level
system—To view this statement in the configuration.
system-control—To add this statement to the configuration.

Related Documentation
- Configuring Locality Bias for a Virtual Chassis on page 144
- Locality Bias in a Virtual Chassis on page 141
- Guidelines for Configuring Locality Bias in a Virtual Chassis on page 143

logical-interface-chassis-redundancy (MX Series Virtual Chassis)

Syntax logical-interface-chassis-redundancy;

Hierarchy Level [edit interfaces ae*number* aggregated-ether-options]

Release Information Statement introduced in Junos OS Release 13.2.

Description For member routers in an MX Series Virtual Chassis, provide chassis redundancy for IP demultiplexing (demux) and VLAN demux subscriber interfaces in aggregated Ethernet bundles configured with targeted distribution. With chassis redundancy, the router assigns the backup link in the aggregated Ethernet bundle to an MPC/MIC module in a member router *other* than the router on which the primary link resides.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring Chassis Redundancy for a Virtual Chassis on page 161

logical-interface-fpc-redundancy (Aggregated Ethernet Subscriber Interfaces)

Syntax logical-interface-fpc-redundancy;

Hierarchy Level [edit interfaces ae*number* aggregated-ether-options]

Release Information Statement introduced in Junos OS Release 11.2.
Statement introduced in Junos OS Release 13.2R2 for EX Series switches.

Description Provide module redundancy for demux subscribers on aggregated Ethernet bundles configured with targeted distribution. Backup links for a subscriber are chosen on a different EQ DPC or MPC from the primary link, based on the link with the fewest number of subscribers among the links on different modules. If all links are on a single module when this is configured, backup links are not provisioned.

By default, link redundancy is provided for the aggregated Ethernet bundle.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- *Configuring Link and Module Redundancy for Demux Subscribers in an Aggregated Ethernet Interface*
- Configuring Module Redundancy for a Virtual Chassis on page 159

member (MX Series Virtual Chassis)

Syntax

```
member member-id {
  role (routing-engine | line-card);
  serial-number serial-number;
}
```

Hierarchy Level [edit virtual-chassis]

Release Information Statement introduced in Junos OS Release 11.2.

Description Configure an MX Series router as a member of a Virtual Chassis configuration. You can configure a maximum of two member routers in an MX Series Virtual Chassis.

Options

member-id—Numeric value that identifies a member router in a Virtual Chassis configuration.

Values: 0 or 1

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

network-services

Syntax network-services (ethernet | enhanced-ethernet | ip | enhanced-ip | lan);

Hierarchy Level [edit chassis]

Release Information Statement introduced before Junos OS Release 8.5.
enhanced-ethernet and **enhanced-ip** options introduced in Junos OS Release 11.4.
limited-ifl-scaling option introduced in Junos OS Release 15.1R3 for MX Series routers.

Description Set the router's network services to a specific mode of operation. On MX240, MX480, and MX960 routers, MPC5E and MPC7E power on only if the network services mode configured is **enhanced-ip** or **enhanced-ethernet**. All the other MPCs work with any of the network services modes.

MX2010 and MX2020 support only **enhanced-ip** and **enhanced-ethernet** network services modes.

Default
- MX80, MX104, MX2010, MX2020—**enhanced-ip**
- MX240, MX480, MX960—**ip**

Options

ethernet—Set the router's network services to Ethernet and use standard, compiled firewall filter format.

enhanced-ethernet—Set the router's network services to enhanced Ethernet and use enhanced mode capabilities. Only Trio MPCs and MS-DPCs are powered on in the chassis.

ip—Set the router's network services to Internet Protocol and use standard, compiled firewall filter format.

enhanced-ip—Set the router's network services to enhanced Internet Protocol and use enhanced mode capabilities. Only Trio MPCs and MS-DPCs are powered on in the chassis. Non-service DPCs do not work with enhanced network services mode options. This feature is enabled by default on MX80, MX104, MX 2010, and 2020 3D Universal Edge Routers.

lan—Set the router's network services to LAN and use standard, compiled firewall filter format. Reboot the system after setting the router's network services to LAN.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- *Network Services Mode Overview*
- *Firewall Filters and Enhanced Network Services Mode Overview* in the *Junos OS Broadband Subscriber Management and Services Library*

- *Configuring Junos OS to Run a Specific Network Services Mode in MX Series Routers*
- Configuring Enhanced IP Network Services for a Virtual Chassis on page 69

no-split-detection (MX Series Virtual Chassis)

Syntax no-split-detection;

Hierarchy Level [edit virtual-chassis]

Release Information Statement introduced in Junos OS Release 11.2.

Description As part of the preprovisioned configuration for an MX Series Virtual Chassis, disable detection of a split in the Virtual Chassis configuration. By default, split detection in the Virtual Chassis is enabled. To maintain the Virtual Chassis configuration in the event of a failure of one of the two member routers, we recommend that you use the **no-split-detection** statement to disable split detection in Virtual Chassis configurations in which you think the backup router is more likely to fail than the link to the backup router.

BEST PRACTICE: We recommend that you use the **no-split-detection** statement to disable split detection for a two-member MX Series Virtual Chassis configuration if you think the backup router is more likely to fail than the Virtual Chassis port links to the backup router. Configuring redundant Virtual Chassis ports on different line cards in each member router reduces the likelihood that all Virtual Chassis port interfaces to the backup router can fail.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Disabling Split Detection in a Virtual Chassis Configuration on page 98
- Split Detection Behavior in a Virtual Chassis on page 38
- Global Roles and Local Roles in a Virtual Chassis on page 30
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

preprovisioned (MX Series Virtual Chassis)

Syntax preprovisioned;

Hierarchy Level [edit virtual-chassis]

Release Information Statement introduced in Junos OS Release 11.2.

Description Enable creation of a Virtual Chassis by means of a preprovisioned configuration.

To configure a Virtual Chassis consisting of MX Series routers, you must create a preprovisioned configuration on the master router in the Virtual Chassis by specifying the serial number, member ID, and role for each router (member chassis) in the Virtual Chassis. When a new member router joins the Virtual Chassis, its serial number is compared against the values specified in the preprovisioned configuration. If the serial number of a joining router does not match any of the configured serial numbers, the software prevents that router from becoming a member of the Virtual Chassis.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

role (MX Series Virtual Chassis)

Syntax role (routing-engine | line-card);

Hierarchy Level [edit virtual-chassis member *member-id*]

Release Information Statement introduced in Junos OS Release 11.2.

Description As part of the preprovisioned configuration for an MX Series Virtual Chassis, assign the role to be performed by each member router in the Virtual Chassis. The preprovisioned configuration permanently associates the member ID and role with the member router's chassis serial number.

Options

routing-engine—Enable the member router to function as the master router or backup router of the Virtual Chassis configuration. The master router maintains the global configuration and state information for both members of the Virtual Chassis, and runs the chassis management processes. The backup router synchronizes with the master router and relays chassis control information, such as line-card presence and alarms, to the master router. If the master router is unavailable, the backup router takes mastership of the Virtual Chassis to preserve routing information and maintain network connectivity without disruption. You must assign the **routing-engine** role to both members of the Virtual Chassis. When the Virtual Chassis is formed, the software runs a mastership election algorithm to determine which of the two member routers functions as the master router and which functions as the backup router of the Virtual Chassis.

line-card—Explicitly configuring a member router with the **line-card** role is *not supported* in the current release. However, when split detection is enabled (the default behavior for a Virtual Chassis) and either the Virtual Chassis ports go down or the backup router fails, the master router takes a **line-card** role. The **line-card** role effectively removes the former master router from the Virtual Chassis configuration until connectivity is restored.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- Virtual Chassis Components Overview on page 24
- Global Roles and Local Roles in a Virtual Chassis on page 30
- Split Detection Behavior in a Virtual Chassis on page 38

sampling-instance

Syntax sampling-instance *instance-name*;

Hierarchy Level [edit chassis fpc *slot-number*],
[edit chassis lcc *number* fpc *slot-number*] (Routing Matrix),
[edit chassis member *member-number* fpc slot *slot-number*]

Release Information Statement introduced in Junos OS Release 9.6.
Support at the **[edit chassis member *member-number* fpc slot *slot-number*]** hierarchy level introduced in Junos OS Release 14.1.
Statement introduced in Junos OS Release 14.1R3 for EX Series switches.

Description Associate a defined sampling instance with a specific FPC, MPC, or DPC for active sampling instances configured at the **[edit forwarding-options sampling]** hierarchy level.

For M120 routers with FEB, this statement must also be configured under **[edit chassis feb *slot-number*]**, in addition to the **[edit forwarding-options sampling]** hierarchy level.

In a two-member MX Series Virtual Chassis, the master router (member 0) uses FPC slot numbers 0 through 11 with no offset; the backup router (member 1) uses FPC slot numbers 12 through 23, with an offset of 12.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *Associating Sampling Instances for Active Flow Monitoring with a Specific FPC, MPC, or DPC*
- Inline Flow Monitoring for Virtual Chassis Overview on page 191

serial-number (MX Series Virtual Chassis)

Syntax serial-number *serial-number*;

Hierarchy Level [edit virtual-chassis member *member-id*]

Release Information Statement introduced in Junos OS Release 11.2.

Description As part of the preprovisioned configuration for an MX Series Virtual Chassis, specify the chassis serial number of each MX Series member router in the Virtual Chassis configuration. If you do not correctly specify a router's serial number in the preprovisioned configuration, the software does not recognize that router as a member of the Virtual Chassis.

Options ***serial-number***—Alphanumeric string that represents the chassis serial number of each member router in the Virtual Chassis configuration. The chassis serial number is located on a label affixed to the side of the MX Series chassis. You can also obtain the router's chassis serial number by issuing the **show chassis hardware** command.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

targeted-distribution (Static Interfaces over Aggregated Ethernet)

Syntax targeted-distribution;

Hierarchy Level [edit interfaces demux0 unit *logical-unit-number*],
[edit interfaces pp0 unit *logical-unit-number*]

Release Information Statement introduced in Junos OS Release 11.2.
Statement introduced in Junos OS Release 13.2R2 for EX Series switches.

Description Configure egress data for a logical interface to be sent across a single member link in an aggregated Ethernet bundle. A backup link is provisioned with CoS scheduling resources in the event that the primary assigned link goes down. The aggregated Ethernet interface must be configured without link protection.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *CoS for PPPoE Subscriber Interfaces Overview*
- *Configuring the Distribution Type for PPPoE Subscribers on Aggregated Ethernet Interfaces*
- *Verifying the Distribution of PPPoE Subscribers in an Aggregated Ethernet Interface*
- Targeted Traffic Distribution on Aggregated Ethernet Interfaces in a Virtual Chassis on page 163
- Configuring Module Redundancy for a Virtual Chassis on page 159
- Configuring Chassis Redundancy for a Virtual Chassis on page 161

traceoptions (MX Series Virtual Chassis)

Syntax

```
traceoptions {
  file filename <files number> <match regular-expression> <no-stamp> <replace> <size
    maximum-file-size> <world-readable | no-world-readable>;
  flag flag <detail> <disable> <receive> <send>;
}
```

Hierarchy Level [edit virtual-chassis]

Release Information Statement introduced in Junos OS Release 11.2.

Description Define tracing operations for the MX Series Virtual Chassis configuration.

Default Tracing operations are disabled.

Options

detail—(Optional) Generate detailed trace information for a flag.

disable—(Optional) Disable a flag.

file *filename*—Name of the file to receive the output of the tracing operation. Enclose the name within quotation marks. All files are placed in the directory **/var/log**.

files *number*—(Optional) Maximum number of trace files. When a trace file named **trace-file** reaches its maximum size, it is renamed **trace-file.0**, then **trace-file.1**, and so on, until the maximum number of trace files is reached. Then the oldest trace file is overwritten. If you specify a maximum number of files, you also must specify a maximum file size with the **size** option.

Range: 2 through 1000

Default: 3 files

flag *flag*—Tracing operation to perform. To specify more than one tracing operation, include multiple flag statements. You can include the following flags:

- **all**—All tracing operations.
- **auto-configuration**—Trace Virtual Chassis ports that have been automatically configured.
- **csn**—Trace Virtual Chassis complete sequence number (CSN) packets.
- **error**—Trace Virtual Chassis errored packets.
- **graceful-restart**—Trace Virtual Chassis graceful restart events.
- **hello**—Trace Virtual Chassis hello packets.
- **krt**—Trace Virtual Chassis kernel routing table (KRT) events.
- **lsp**—Trace Virtual Chassis link-state packets.
- **lsp-generation**—Trace Virtual Chassis link-state packet generation.

- **me**—Trace Virtual Chassis mastership election (ME) events.
- **normal**—Trace normal events.
- **packets**—Trace Virtual Chassis packets.
- **parse**—Trace reading of the configuration.
- **psn**—Trace partial sequence number (PSN) packets.
- **route**—Trace Virtual Chassis routing information.
- **spf**—Trace Virtual Chassis shortest-path-first (SPF) events.
- **state**—Trace Virtual Chassis state transitions.
- **task**—Trace Virtual Chassis task operations.

match ***regular-expression***—(Optional) Refine the output to include lines that contain the regular expression.

no-stamp—(Optional) Do not place a timestamp on any trace file.

no-world-readable—(Optional) Restrict file access to the user who created the file.

receive—(Optional) Trace received packets.

replace—(Optional) Replace a trace file instead of appending information to it.

send—(Optional) Trace transmitted packets.

size ***maximum-file-size***—(Optional) Maximum size of each trace file. By default, the number entered is treated as bytes. Alternatively, you can include a suffix to the number to indicate kilobytes (KB), megabytes (MB), or gigabytes (GB). If you specify a maximum file size, you also must specify a maximum number of trace files with the **files** option.

Syntax: ***size*****k** to specify KB, ***size*****m** to specify MB, or ***size*****g** to specify GB

Range: 10240 through 1073741824

world-readable—(Optional) Enable unrestricted file access.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Tracing Virtual Chassis Operations for MX Series 3D Universal Edge Routers on page 224
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

virtual-chassis (MX Series Virtual Chassis)

Syntax

```
virtual-chassis {
  aliases {
    serial-number serial-number;
  }
  graceful-restart {
    disable;
  }
  heartbeat-address (ip-address | ipv6-address);
  heartbeat-timeout seconds;
  heartbeat-tos seconds;
  heartbeat-tos value;
  locality-bias;
  member member-id {
    role (routing-engine | line-card);
    serial-number serial-number;
  }
  no-split-detection;
  preprovisioned;
  traceoptions {
    file filename <files number> <match regular-expression> <no-stamp> <replace> <size
      maximum-file-size> <world-readable | no-world-readable>;
    flag flag <detail> <disable> <receive> <send>;
  }
}
```

Hierarchy Level [edit]

Release Information Statement introduced in Junos OS Release 11.2.

Description Create a Virtual Chassis configuration for MX Series routers.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet on page 196
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets on page 207
- Configuring Preprovisioned Member Information for a Virtual Chassis on page 66
- Configuring Locality Bias for a Virtual Chassis on page 144

CHAPTER 13

Operational Commands

- clear virtual-chassis heartbeat (MX Series Virtual Chassis)
- request system software in-service-upgrade (MX Series 3D Universal Edge Routers and EX9200 Switches)
- request virtual-chassis member-id delete (MX Series Virtual Chassis)
- request virtual-chassis member-id set
- request virtual-chassis routing-engine master switch
- request virtual-chassis vc-port delete (MX Series Virtual Chassis)
- request virtual-chassis vc-port set (MX Series Virtual Chassis)
- show chassis network-services
- show interfaces vcp
- show services accounting status
- show virtual-chassis active-topology (MX Series Virtual Chassis)
- show virtual-chassis device-topology (MX Series Virtual Chassis)
- show virtual-chassis heartbeat (MX Series Virtual Chassis)
- show virtual-chassis protocol adjacency (MX Series Virtual Chassis)
- show virtual-chassis protocol database (MX Series Virtual Chassis)
- show virtual-chassis protocol interface (MX Series Virtual Chassis)
- show virtual-chassis protocol route (MX Series Virtual Chassis)
- show virtual-chassis protocol statistics (MX Series Virtual Chassis)
- show virtual-chassis status (MX Series Virtual Chassis)
- show virtual-chassis vc-port (MX Series Virtual Chassis)

clear virtual-chassis heartbeat (MX Series Virtual Chassis)

Syntax clear virtual-chassis heartbeat
<(all-members | local | member *member-id*)>

Release Information Command introduced in Junos OS Release 14.1.

Description Clear statistics counter and timestamp fields associated with a heartbeat packet connection on both member routers in an MX Series Virtual Chassis. You can issue the **clear virtual-chassis heartbeat** command from the console of either member router in the Virtual Chassis.

NOTE: When you clear heartbeat statistics, the connection time is preserved to retain a record of previous heartbeat connectivity.

Options

none—Clear heartbeat statistics for both member routers in an MX Series Virtual Chassis.

all-members—(Optional) Clear heartbeat statistics for both member routers in an MX Series Virtual Chassis. This is the default behavior if no options are specified.

local—(Optional) Clear heartbeat statistics for the member router from which you are issuing the command.

member *member-id*—(Optional) Clear heartbeat statistics for the specified member router. Replace ***member-id*** with the value 0 or 1.

Required Privilege Level clear

Related Documentation

- Verifying and Managing the Virtual Chassis Heartbeat Connection on page 190
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet on page 196
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets on page 207

List of Sample Output clear virtual-chassis heartbeat on page 253

Output Fields Table 27 on page 253 lists the output fields for the **clear virtual-chassis heartbeat** command. Output fields are listed in the approximate order in which they appear.

Table 27: clear virtual-chassis heartbeat Output Fields

Field Name	Field Description
member*n*	Member ID of the Virtual Chassis member router for which output is displayed.
heartbeat statistics cleared	Confirmation that heartbeat statistics are cleared on the specified member router.

Sample Output

clear virtual-chassis heartbeat

```
{master:member0-re0}

user@host> clear virtual-chassis heartbeat
member0:
--------------------------------------------------------------------------
heartbeat statistics cleared

member1:
--------------------------------------------------------------------------
heartbeat statistics cleared
```

request system software in-service-upgrade (MX Series 3D Universal Edge Routers and EX9200 Switches)

Syntax request system software in-service-upgrade *package-name*
<no-copy>
<no-old-master-upgrade>
<reboot>
<unlink>

Release Information Command introduced in Junos OS Release 11.2.
Command introduced in Junos OS Release 14.1 for MX Series Virtual Chassis.
Command introduced in Junos OS Release 14.2 for EX Series switches.

Description Perform a unified in-service software upgrade (unified ISSU). Unified ISSU enables you to upgrade from one Junos OS release to another with no disruption on the control plane and with minimal disruption of traffic. Unified ISSU is supported only by dual Routing Engine platforms. In addition, graceful Routing Engine switchover (GRES) and nonstop active routing (NSR) must be enabled.

Options *package-name*—Location from which the software package or bundle is to be installed. For example:

- */var/tmp/package-name*— For a software package or bundle that is being installed from a local directory on the router.
- *protocol://hostname/pathname/package-name*—For a software package or bundle that is to be downloaded and installed from a remote location. Replace ***protocol*** with one of the following:
  - **ftp**—File Transfer Protocol
  - **http**—Hypertext Transfer Protocol
  - **scp**—Secure copy (available only for Canada and U.S. version)

no-copy—(Optional) When the **no-copy** option is included, copies of package files are not saved on the Packet Forwarding Engine.

The **no-copy** option is not available for an MX Series Virtual Chassis or an EX9200 Virtual Chassis.

no-old-master-upgrade—(Optional) When the **no-old-master-upgrade** option is included, after the backup Routing Engine is rebooted with the new software package and a switchover occurs to make it the new master Routing Engine, the former master (new backup) Routing Engine is not upgraded to the new software. In this case, you must manually upgrade the former master (new backup) Routing Engine. If you do not include the **no-old-master-upgrade** option, the system automatically upgrades the former master Routing Engine.

The **no-old-master-upgrade** option is not available for an MX Series Virtual Chassis or an EX9200 Virtual Chassis.

reboot—(Optional) When the **reboot** option is included, the former master (new backup) Routing Engine is automatically rebooted after being upgraded to the new software. When the **reboot** option is not included, you must manually reboot the former master (new backup) Routing Engine using the **request system reboot** command.

The **reboot** option is accepted but ignored for an MX Series Virtual Chassis or an EX9200 Virtual Chassis. A unified ISSU in an MX Series Virtual Chassis or EX9200 Virtual Chassis always reboots all Routing Engines in the member routers or switches.

unlink—(Optional) When the **unlink** option is included, the package is removed from **/var/home** whether the installation is successful or unsuccessful.

The **unlink** option is not available for an MX Series Virtual Chassis or an EX9200 Virtual Chassis.

Additional Information

The following conditions apply to unified ISSUs:

- Unified ISSUs are supported on MX Series 3D Universal Edge Routers and EX9200 switches.
- Unsupported PICs (on EX9200, PICs are known as "line cards") are restarted during a unified ISSU. For information about supported PICs, see the *Junos OS High Availability Library for Routing Devices*. For information about supported EX9200 line cards, see *Unified ISSU System Requirements*.
- Unsupported protocols will experience packet loss during a unified ISSU. For information about supported protocols, see the *Junos OS High Availability Library for Routing Devices* or, for EX9200, see *Unified ISSU System Requirements*.
- During a unified ISSU, you cannot bring any PICs online or offline.

For more information, see the *Junos OS High Availability Library for Routing Devices* or the *High Availability Feature Guide for EX9200 Switches*.

Required Privilege Level

view

Related Documentation

- *request system software abort*
- *show chassis in-service-upgrade*

List of Sample Output

request system software in-service-upgrade reboot on page 255
request system software in-service-upgrade (MX Series Virtual Chassis) on page 266

Output Fields

When you enter this command, you are provided feedback about the status of your request.

Sample Output

request system software in-service-upgrade reboot

```
{master}
```

```
user@host> request system software in-service-upgrade
/var/tmp/jinstall-11.2B2.1-domestic-signed.tgz reboot
Chassis ISSU Check Done
ISSU: Validating Image
Checking compatibility with configuration
Initializing...
Using jbase-11.2B1.5
Verified manifest signed by PackageProduction_11_2_0
Verified jbase-11.2B1.5 signed by PackageProduction_11_2_0
Using /var/tmp/jinstall-11.2B2.1-domestic-signed.tgz
Verified jinstall-11.2B2.1-domestic.tgz signed by PackageProduction_11_2_0
Using jinstall-11.2B2.1-domestic.tgz
Using jbundle-11.2B2.1-domestic.tgz
Checking jbundle requirements on /
Using jbase-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jbase-11.2B2.1 signed by PackageProduction_11_2_0
Using /var/validate/chroot/tmp/jbundle/jboot-11.2B2.1.tgz
Using jcrypto-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jcrypto-11.2B2.1 signed by PackageProduction_11_2_0
Using jdocs-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jdocs-11.2B2.1 signed by PackageProduction_11_2_0
Using jkernel-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jkernel-11.2B2.1 signed by PackageProduction_11_2_0
Using jpfe-11.2B2.1.tgz
Using jroute-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jroute-11.2B2.1 signed by PackageProduction_11_2_0
Using jruntime-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jruntime-11.2B2.1 signed by PackageProduction_11_2_0
Using jservices-11.2B2.1.tgz
Auto-deleting old jservices-voice ...
Removing /opt/sdk/service-packages/jservices-voice ...
Removing jservices-voice-bsg-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-voice ...
Verified jservices-voice-bsg-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /var/sw/pkg ...
Creating /opt/sdk/service-packages/jservices-voice ...
Storing jservices-voice-bsg-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-voice/jservices-voice-bsg ->
/var/sw/pkg/jservices-voice-bsg-11.2B2.1.tgz...
Auto-deleting old jservices-bgf ...
Removing /opt/sdk/service-packages/jservices-bgf ...
Removing jservices-bgf-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-bgf ...
Verified jservices-bgf-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-bgf ...
Storing jservices-bgf-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-bgf/jservices-bgf-pic ->
/var/sw/pkg/jservices-bgf-pic-11.2B2.1.tgz...
Auto-deleting old jservices-aacl ...
Removing /opt/sdk/service-packages/jservices-aacl ...
Removing jservices-aacl-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-aacl ...
```

```
Verified jservices-aacl-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-aacl ...
Storing jservices-aacl-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-aacl/jservices-aacl-pic ->
/var/sw/pkg/jservices-aacl-pic-11.2B2.1.tgz...
Auto-deleting old jservices-llpdf ...
Removing /opt/sdk/service-packages/jservices-llpdf ...
Removing jservices-llpdf-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-llpdf ...
Verified jservices-llpdf-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-llpdf ...
Storing jservices-llpdf-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-llpdf/jservices-llpdf-pic ->
/var/sw/pkg/jservices-llpdf-pic-11.2B2.1.tgz...
Auto-deleting old jservices-ptsp ...
Removing /opt/sdk/service-packages/jservices-ptsp ...
Removing jservices-ptsp-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-ptsp ...
Verified jservices-ptsp-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-ptsp ...
Storing jservices-ptsp-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-ptsp/jservices-ptsp-pic ->
/var/sw/pkg/jservices-ptsp-pic-11.2B2.1.tgz...
Auto-deleting old jservices-sfw ...
Removing /opt/sdk/service-packages/jservices-sfw ...
Removing jservices-sfw-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-sfw ...
Verified jservices-sfw-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-sfw ...
Storing jservices-sfw-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-sfw/jservices-sfw-pic ->
/var/sw/pkg/jservices-sfw-pic-11.2B2.1.tgz...
Auto-deleting old jservices-nat ...
Removing /opt/sdk/service-packages/jservices-nat ...
Removing jservices-nat-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-nat ...
Verified jservices-nat-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-nat ...
Storing jservices-nat-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-nat/jservices-nat-pic ->
/var/sw/pkg/jservices-nat-pic-11.2B2.1.tgz...
Auto-deleting old jservices-alg ...
Removing /opt/sdk/service-packages/jservices-alg ...
Removing jservices-alg-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-alg ...
Verified jservices-alg-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-alg ...
Storing jservices-alg-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-alg/jservices-alg-pic ->
/var/sw/pkg/jservices-alg-pic-11.2B2.1.tgz...
Auto-deleting old jservices-cpcd ...
Removing /opt/sdk/service-packages/jservices-cpcd ...
Removing jservices-cpcd-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-cpcd ...
Verified jservices-cpcd-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
```

```
Creating /opt/sdk/service-packages/jservices-cpcd ...
Storing jservices-cpcd-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-cpcd/jservices-cpcd-pic ->
/var/sw/pkg/jservices-cpcd-pic-11.2B2.1.tgz...
Auto-deleting old jservices-rpm ...
Removing /opt/sdk/service-packages/jservices-rpm ...
Removing jservices-rpm-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-rpm ...
Verified jservices-rpm-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-rpm ...
Storing jservices-rpm-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-rpm/jservices-rpm-pic ->
/var/sw/pkg/jservices-rpm-pic-11.2B2.1.tgz...
Auto-deleting old jservices-hcm ...
Removing /opt/sdk/service-packages/jservices-hcm ...
Removing jservices-hcm-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-hcm ...
Verified jservices-hcm-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-hcm ...
Storing jservices-hcm-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-hcm/jservices-hcm-pic ->
/var/sw/pkg/jservices-hcm-pic-11.2B2.1.tgz...
Auto-deleting old jservices-appid ...
Removing /opt/sdk/service-packages/jservices-appid ...
Removing jservices-appid-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-appid ...
Verified jservices-appid-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-appid ...
Storing jservices-appid-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-appid/jservices-appid-pic ->
/var/sw/pkg/jservices-appid-pic-11.2B2.1.tgz...
Auto-deleting old jservices-idp ...
Removing /opt/sdk/service-packages/jservices-idp ...
Removing jservices-idp-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-idp ...
Verified jservices-idp-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-idp ...
Storing jservices-idp-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-idp/jservices-idp-pic ->
/var/sw/pkg/jservices-idp-pic-11.2B2.1.tgz...
Using jservices-crypto-11.2B2.1.tgz
Auto-deleting old jservices-crypto-base ...
Removing /opt/sdk/service-packages/jservices-crypto-base ...
Removing jservices-crypto-base-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-crypto-base ...
Verified jservices-crypto-base-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-crypto-base ...
Storing jservices-crypto-base-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-crypto-base/jservices-crypto-base-pic
-> /var/sw/pkg/jservices-crypto-base-pic-11.2B2.1.tgz...
Auto-deleting old jservices-ssl ...
Removing /opt/sdk/service-packages/jservices-ssl ...
Removing jservices-ssl-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-ssl ...
Verified jservices-ssl-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
```

```
Creating /opt/sdk/service-packages/jservices-ssl ...
Storing jservices-ssl-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-ssl/jservices-ssl-pic ->
/var/sw/pkg/jservices-ssl-pic-11.2B2.1.tgz...
Hardware Database regeneration succeeded
Validating against /config/juniper.conf.gz
mgd: commit complete
Validation succeeded
ISSU: Preparing Backup RE
Pushing bundle to re1
NOTICE: Validating configuration against jinstall-11.2B2.1-domestic-signed.tgz.
NOTICE: Use the 'no-validate' option to skip this if desired.
Checking compatibility with configuration
Initializing...
Using jbase-11.2B1.5
Verified manifest signed by PackageProduction_11_2_0
Verified jbase-11.2B1.5 signed by PackageProduction_11_2_0
Using /var/tmp/jinstall-11.2B2.1-domestic-signed.tgz
Verified jinstall-11.2B2.1-domestic.tgz signed by PackageProduction_11_2_0
Using jinstall-11.2B2.1-domestic.tgz
Using jbundle-11.2B2.1-domestic.tgz
Checking jbundle requirements on /
Using jbase-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jbase-11.2B2.1 signed by PackageProduction_11_2_0
Using /var/validate/chroot/tmp/jbundle/jboot-11.2B2.1.tgz
Using jcrypto-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jcrypto-11.2B2.1 signed by PackageProduction_11_2_0
Using jdocs-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jdocs-11.2B2.1 signed by PackageProduction_11_2_0
Using jkernel-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jkernel-11.2B2.1 signed by PackageProduction_11_2_0
Using jpfe-11.2B2.1.tgz
Using jroute-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jroute-11.2B2.1 signed by PackageProduction_11_2_0
Using jruntime-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jruntime-11.2B2.1 signed by PackageProduction_11_2_0
Using jservices-11.2B2.1.tgz
Auto-deleting old jservices-voice ...
Removing /opt/sdk/service-packages/jservices-voice ...
Removing jservices-voice-bsg-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-voice ...
Verified jservices-voice-bsg-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /var/sw/pkg ...
Creating /opt/sdk/service-packages/jservices-voice ...
Storing jservices-voice-bsg-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-voice/jservices-voice-bsg ->
/var/sw/pkg/jservices-voice-bsg-11.2B2.1.tgz...
Auto-deleting old jservices-bgf ...
Removing /opt/sdk/service-packages/jservices-bgf ...
Removing jservices-bgf-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-bgf ...
Verified jservices-bgf-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-bgf ...
```

```
Storing jservices-bgf-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-bgf/jservices-bgf-pic ->
/var/sw/pkg/jservices-bgf-pic-11.2B2.1.tgz...
Auto-deleting old jservices-aacl ...
Removing /opt/sdk/service-packages/jservices-aacl ...
Removing jservices-aacl-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-aacl ...
Verified jservices-aacl-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-aacl ...
Storing jservices-aacl-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-aacl/jservices-aacl-pic ->
/var/sw/pkg/jservices-aacl-pic-11.2B2.1.tgz...
Auto-deleting old jservices-llpdf ...
Removing /opt/sdk/service-packages/jservices-llpdf ...
Removing jservices-llpdf-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-llpdf ...
Verified jservices-llpdf-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-llpdf ...
Storing jservices-llpdf-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-llpdf/jservices-llpdf-pic ->
/var/sw/pkg/jservices-llpdf-pic-11.2B2.1.tgz...
Auto-deleting old jservices-ptsp ...
Removing /opt/sdk/service-packages/jservices-ptsp ...
Removing jservices-ptsp-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-ptsp ...
Verified jservices-ptsp-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-ptsp ...
Storing jservices-ptsp-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-ptsp/jservices-ptsp-pic ->
/var/sw/pkg/jservices-ptsp-pic-11.2B2.1.tgz...
Auto-deleting old jservices-sfw ...
Removing /opt/sdk/service-packages/jservices-sfw ...
Removing jservices-sfw-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-sfw ...
Verified jservices-sfw-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-sfw ...
Storing jservices-sfw-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-sfw/jservices-sfw-pic ->
/var/sw/pkg/jservices-sfw-pic-11.2B2.1.tgz...
Auto-deleting old jservices-nat ...
Removing /opt/sdk/service-packages/jservices-nat ...
Removing jservices-nat-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-nat ...
Verified jservices-nat-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-nat ...
Storing jservices-nat-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-nat/jservices-nat-pic ->
/var/sw/pkg/jservices-nat-pic-11.2B2.1.tgz...
Auto-deleting old jservices-alg ...
Removing /opt/sdk/service-packages/jservices-alg ...
Removing jservices-alg-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-alg ...
Verified jservices-alg-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-alg ...
Storing jservices-alg-pic-11.2B2.1.tgz in /var/sw/pkg ...
```

```
Link: /opt/sdk/service-packages/jservices-alg/jservices-alg-pic ->
/var/sw/pkg/jservices-alg-pic-11.2B2.1.tgz...
Auto-deleting old jservices-cpcd ...
Removing /opt/sdk/service-packages/jservices-cpcd ...
Removing jservices-cpcd-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-cpcd ...
Verified jservices-cpcd-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-cpcd ...
Storing jservices-cpcd-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-cpcd/jservices-cpcd-pic ->
/var/sw/pkg/jservices-cpcd-pic-11.2B2.1.tgz...
Auto-deleting old jservices-rpm ...
Removing /opt/sdk/service-packages/jservices-rpm ...
Removing jservices-rpm-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-rpm ...
Verified jservices-rpm-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-rpm ...
Storing jservices-rpm-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-rpm/jservices-rpm-pic ->
/var/sw/pkg/jservices-rpm-pic-11.2B2.1.tgz...
Auto-deleting old jservices-hcm ...
Removing /opt/sdk/service-packages/jservices-hcm ...
Removing jservices-hcm-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-hcm ...
Verified jservices-hcm-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-hcm ...
Storing jservices-hcm-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-hcm/jservices-hcm-pic ->
/var/sw/pkg/jservices-hcm-pic-11.2B2.1.tgz...
Auto-deleting old jservices-appid ...
Removing /opt/sdk/service-packages/jservices-appid ...
Removing jservices-appid-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-appid ...
Verified jservices-appid-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-appid ...
Storing jservices-appid-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-appid/jservices-appid-pic ->
/var/sw/pkg/jservices-appid-pic-11.2B2.1.tgz...
Auto-deleting old jservices-idp ...
Removing /opt/sdk/service-packages/jservices-idp ...
Removing jservices-idp-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-idp ...
Verified jservices-idp-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-idp ...
Storing jservices-idp-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-idp/jservices-idp-pic ->
/var/sw/pkg/jservices-idp-pic-11.2B2.1.tgz...
Using jservices-crypto-11.2B2.1.tgz
Auto-deleting old jservices-crypto-base ...
Removing /opt/sdk/service-packages/jservices-crypto-base ...
Removing jservices-crypto-base-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-crypto-base ...
Verified jservices-crypto-base-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-crypto-base ...
Storing jservices-crypto-base-pic-11.2B2.1.tgz in /var/sw/pkg ...
```

```
Link: /opt/sdk/service-packages/jservices-crypto-base/jservices-crypto-base-pic
-> /var/sw/pkg/jservices-crypto-base-pic-11.2B2.1.tgz...
Auto-deleting old jservices-ssl ...
Removing /opt/sdk/service-packages/jservices-ssl ...
Removing jservices-ssl-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-ssl ...
Verified jservices-ssl-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-ssl ...
Storing jservices-ssl-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-ssl/jservices-ssl-pic ->
/var/sw/pkg/jservices-ssl-pic-11.2B2.1.tgz...
Hardware Database regeneration succeeded
Validating against /config/juniper.conf.gz
mgd: commit complete
Validation succeeded
Installing package '/var/tmp/jinstall-11.2B2.1-domestic-signed.tgz' ...
Verified jinstall-11.2B2.1-domestic.tgz signed by PackageProduction_11_2_0
Adding jinstall...
Verified manifest signed by PackageProduction_11_2_0

WARNING:     This package will load JUNOS 11.2B2.1 software.
WARNING:     It will save JUNOS configuration files, and SSH keys
WARNING:     (if configured), but erase all other files and information
WARNING:     stored on this machine.  It will attempt to preserve dumps
WARNING:     and log files, but this can not be guaranteed.  This is the
WARNING:     pre-installation stage and all the software is loaded when
WARNING:     you reboot the system.

Saving the config files ...
NOTICE: uncommitted changes have been saved in
/var/db/config/juniper.conf.pre-install
Installing the bootstrap installer ...

WARNING:     A REBOOT IS REQUIRED TO LOAD THIS SOFTWARE CORRECTLY. Use the
WARNING:     'request system reboot' command when software installation is
WARNING:     complete. To abort the installation, do not reboot your system,
WARNING:     instead use the 'request system software delete jinstall'
WARNING:     command as soon as this operation completes.

Saving package file in /var/sw/pkg/jinstall-11.2B2.1-domestic-signed.tgz ...
Saving state for rollback ...
Backup upgrade done
Rebooting Backup RE

Rebooting re1
ISSU: Backup RE Prepare Done
Waiting for Backup RE reboot
GRES operational
Initiating Chassis In-Service-Upgrade
Chassis ISSU Started
ISSU: Preparing Daemons
ISSU: Daemons Ready for ISSU
ISSU: Starting Upgrade for FRUs
ISSU: Preparing for Switchover
ISSU: Ready for Switchover
Checking In-Service-Upgrade status
  Item           Status                  Reason
  FPC 1          Online (ISSU)
  FPC 4          Online (ISSU)
  FPC 8          Online (ISSU)
```



```
  FPC 10         Online (ISSU)
Resolving mastership...
Complete. The other routing engine becomes the master.
ISSU: RE switchover Done
ISSU: Upgrading Old Master RE
NOTICE: Validating configuration against jinstall-11.2B2.1-domestic-signed.tgz.
NOTICE: Use the 'no-validate' option to skip this if desired.
Checking compatibility with configuration
Initializing...
Using jbase-11.2B1.5
Verified manifest signed by PackageProduction_11_2_0
Verified jbase-11.2B1.5 signed by PackageProduction_11_2_0
Using /var/tmp/jinstall-11.2B2.1-domestic-signed.tgz
Verified jinstall-11.2B2.1-domestic.tgz signed by PackageProduction_11_2_0
Using jinstall-11.2B2.1-domestic.tgz
Using jbundle-11.2B2.1-domestic.tgz
Checking jbundle requirements on /
Using jbase-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jbase-11.2B2.1 signed by PackageProduction_11_2_0
Using /var/validate/chroot/tmp/jbundle/jboot-11.2B2.1.tgz
Using jcrypto-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jcrypto-11.2B2.1 signed by PackageProduction_11_2_0
Using jdocs-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jdocs-11.2B2.1 signed by PackageProduction_11_2_0
Using jkernel-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jkernel-11.2B2.1 signed by PackageProduction_11_2_0
Using jpfe-11.2B2.1.tgz
Using jroute-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jroute-11.2B2.1 signed by PackageProduction_11_2_0
Using jruntime-11.2B2.1.tgz
Verified manifest signed by PackageProduction_11_2_0
Verified jruntime-11.2B2.1 signed by PackageProduction_11_2_0
Using jservices-11.2B2.1.tgz
Auto-deleting old jservices-voice ...
Removing /opt/sdk/service-packages/jservices-voice ...
Removing jservices-voice-bsg-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-voice ...
Verified jservices-voice-bsg-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /var/sw/pkg ...
Creating /opt/sdk/service-packages/jservices-voice ...
Storing jservices-voice-bsg-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-voice/jservices-voice-bsg ->
/var/sw/pkg/jservices-voice-bsg-11.2B2.1.tgz...
Auto-deleting old jservices-bgf ...
Removing /opt/sdk/service-packages/jservices-bgf ...
Removing jservices-bgf-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-bgf ...
Verified jservices-bgf-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-bgf ...
Storing jservices-bgf-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-bgf/jservices-bgf-pic ->
/var/sw/pkg/jservices-bgf-pic-11.2B2.1.tgz...
Auto-deleting old jservices-aacl ...
Removing /opt/sdk/service-packages/jservices-aacl ...
```

```
Removing jservices-aacl-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-aacl ...
Verified jservices-aacl-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-aacl ...
Storing jservices-aacl-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-aacl/jservices-aacl-pic ->
/var/sw/pkg/jservices-aacl-pic-11.2B2.1.tgz...
Auto-deleting old jservices-llpdf ...
Removing /opt/sdk/service-packages/jservices-llpdf ...
Removing jservices-llpdf-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-llpdf ...
Verified jservices-llpdf-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-llpdf ...
Storing jservices-llpdf-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-llpdf/jservices-llpdf-pic ->
/var/sw/pkg/jservices-llpdf-pic-11.2B2.1.tgz...
Auto-deleting old jservices-ptsp ...
Removing /opt/sdk/service-packages/jservices-ptsp ...
Removing jservices-ptsp-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-ptsp ...
Verified jservices-ptsp-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-ptsp ...
Storing jservices-ptsp-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-ptsp/jservices-ptsp-pic ->
/var/sw/pkg/jservices-ptsp-pic-11.2B2.1.tgz...
Auto-deleting old jservices-sfw ...
Removing /opt/sdk/service-packages/jservices-sfw ...
Removing jservices-sfw-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-sfw ...
Verified jservices-sfw-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-sfw ...
Storing jservices-sfw-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-sfw/jservices-sfw-pic ->
/var/sw/pkg/jservices-sfw-pic-11.2B2.1.tgz...
Auto-deleting old jservices-nat ...
Removing /opt/sdk/service-packages/jservices-nat ...
Removing jservices-nat-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-nat ...
Verified jservices-nat-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-nat ...
Storing jservices-nat-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-nat/jservices-nat-pic ->
/var/sw/pkg/jservices-nat-pic-11.2B2.1.tgz...
Auto-deleting old jservices-alg ...
Removing /opt/sdk/service-packages/jservices-alg ...
Removing jservices-alg-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-alg ...
Verified jservices-alg-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-alg ...
Storing jservices-alg-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-alg/jservices-alg-pic ->
/var/sw/pkg/jservices-alg-pic-11.2B2.1.tgz...
Auto-deleting old jservices-cpcd ...
Removing /opt/sdk/service-packages/jservices-cpcd ...
Removing jservices-cpcd-pic-11.2B1.5.tgz from /var/sw/pkg ...
```



```
Notifying mspd ...
Installing new jservices-cpcd ...
Verified jservices-cpcd-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-cpcd ...
Storing jservices-cpcd-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-cpcd/jservices-cpcd-pic ->
/var/sw/pkg/jservices-cpcd-pic-11.2B2.1.tgz...
Auto-deleting old jservices-rpm ...
Removing /opt/sdk/service-packages/jservices-rpm ...
Removing jservices-rpm-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-rpm ...
Verified jservices-rpm-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-rpm ...
Storing jservices-rpm-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-rpm/jservices-rpm-pic ->
/var/sw/pkg/jservices-rpm-pic-11.2B2.1.tgz...
Auto-deleting old jservices-hcm ...
Removing /opt/sdk/service-packages/jservices-hcm ...
Removing jservices-hcm-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-hcm ...
Verified jservices-hcm-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-hcm ...
Storing jservices-hcm-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-hcm/jservices-hcm-pic ->
/var/sw/pkg/jservices-hcm-pic-11.2B2.1.tgz...
Auto-deleting old jservices-appid ...
Removing /opt/sdk/service-packages/jservices-appid ...
Removing jservices-appid-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-appid ...
Verified jservices-appid-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-appid ...
Storing jservices-appid-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-appid/jservices-appid-pic ->
/var/sw/pkg/jservices-appid-pic-11.2B2.1.tgz...
Auto-deleting old jservices-idp ...
Removing /opt/sdk/service-packages/jservices-idp ...
Removing jservices-idp-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-idp ...
Verified jservices-idp-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-idp ...
Storing jservices-idp-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-idp/jservices-idp-pic ->
/var/sw/pkg/jservices-idp-pic-11.2B2.1.tgz...
Using jservices-crypto-11.2B2.1.tgz
Auto-deleting old jservices-crypto-base ...
Removing /opt/sdk/service-packages/jservices-crypto-base ...
Removing jservices-crypto-base-pic-11.2B1.5.tgz from /var/sw/pkg ...
Notifying mspd ...
Installing new jservices-crypto-base ...
Verified jservices-crypto-base-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-crypto-base ...
Storing jservices-crypto-base-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-crypto-base/jservices-crypto-base-pic
-> /var/sw/pkg/jservices-crypto-base-pic-11.2B2.1.tgz...
Auto-deleting old jservices-ssl ...
Removing /opt/sdk/service-packages/jservices-ssl ...
Removing jservices-ssl-pic-11.2B1.5.tgz from /var/sw/pkg ...
```

```
Notifying mspd ...
Installing new jservices-ssl ...
Verified jservices-ssl-pic-11.2B2.1.tgz signed by PackageProduction_11_2_0
Creating /opt/sdk/service-packages/jservices-ssl ...
Storing jservices-ssl-pic-11.2B2.1.tgz in /var/sw/pkg ...
Link: /opt/sdk/service-packages/jservices-ssl/jservices-ssl-pic ->
/var/sw/pkg/jservices-ssl-pic-11.2B2.1.tgz...
Hardware Database regeneration succeeded
Validating against /config/juniper.conf.gz
mgd: commit complete
Validation succeeded
Installing package '/var/tmp/jinstall-11.2B2.1-domestic-signed.tgz' ...
Verified jinstall-11.2B2.1-domestic.tgz signed by PackageProduction_11_2_0
Adding jinstall...
Verified manifest signed by PackageProduction_11_2_0

WARNING:     This package will load JUNOS 11.2B2.1 software.
WARNING:     It will save JUNOS configuration files, and SSH keys
WARNING:     (if configured), but erase all other files and information
WARNING:     stored on this machine.  It will attempt to preserve dumps
WARNING:     and log files, but this can not be guaranteed.  This is the
WARNING:     pre-installation stage and all the software is loaded when
WARNING:     you reboot the system.

Saving the config files ...
NOTICE: uncommitted changes have been saved in
/var/db/config/juniper.conf.pre-install
Installing the bootstrap installer ...

WARNING:     A REBOOT IS REQUIRED TO LOAD THIS SOFTWARE CORRECTLY. Use the
WARNING:     'request system reboot' command when software installation is
WARNING:     complete. To abort the installation, do not reboot your system,
WARNING:     instead use the 'request system software delete jinstall'
WARNING:     command as soon as this operation completes.

Saving package file in /var/sw/pkg/jinstall-11.2B2.1-domestic-signed.tgz ...
Saving state for rollback ...
ISSU: Old Master Upgrade Done
ISSU: IDLE
Shutdown NOW!
Reboot consistency check bypassed - jinstall 11.2B2.1 will complete installation
 upon reboot
[pid 66780]

*** FINAL System shutdown message from user@host> ***
System going down IMMEDIATELY
```

request system software in-service-upgrade (MX Series Virtual Chassis)

```
{master:member0-re0}

user@host> request system software in-service-upgrade
jinstall-14.1-20140114.2-domestic-signed.tgz
[Jan 30 10:45:32]:ISSU: IDLE

Beginning in-service-upgrade at Jan 30, 2014; 10:45:34
[Jan 30 10:45:34]:ISSU: Validating Image
Validating VC readiness...
Validating required configuration...
Validating release compatibility...
Validation successful
```

```
Initiating chassis in-service-upgrade
[Jan 30 10:46:56]:ISSU: Preparing LCC Backup REs
Copying new release to all RE's
Pushing bundle to member0-re0
Pushing bundle to member1-re0
Pushing bundle to member1-re1
[Jan 30 10:51:11]:ISSU: Preparing Backup RE
Arming new release on all RE's
member0-re0:
--------------------------------------------------------------------------
Installing package
'/var/tmp/jinstall-14.1-20140114_ib_14_1_psd.1-domestic-signed.tgz' ...
Verified jinstall-14.1-20140114_ib_14_1_psd.1-domestic.tgz signed by
PackageDevelopmentEc_2014
Adding jinstall...

WARNING:     The software that is being installed has limited support.
WARNING:     Run 'file show /etc/notices/unsupported.txt' for details.

veriexec: accepting signer: PackageDevelopmentEc_2014
Verified manifest signed by PackageDevelopmentEc_2014

WARNING:     This package will load JUNOS 14.1-20140114_ib_14_1_psd.1 software.
WARNING:     It will save JUNOS configuration files, and SSH keys
WARNING:     (if configured), but erase all other files and information
WARNING:     stored on this machine.  It will attempt to preserve dumps
WARNING:     and log files, but this can not be guaranteed.  This is the
WARNING:     pre-installation stage and all the software is loaded when
WARNING:     you reboot the system.

Saving the config files ...
NOTICE: uncommitted changes have been saved in
/var/db/config/juniper.conf.pre-install
Installing the bootstrap installer ...

WARNING:     A REBOOT IS REQUIRED TO LOAD THIS SOFTWARE CORRECTLY. Use the
WARNING:     'request system reboot' command when software installation is
WARNING:     complete. To abort the installation, do not reboot your system,
WARNING:     instead use the 'request system software delete jinstall'
WARNING:     command as soon as this operation completes.

Saving package file in
/var/sw/pkg/jinstall-14.1-20140114_ib_14_1_psd.1-domestic-signed.tgz ...
Saving state for rollback ...

member1-re0:
--------------------------------------------------------------------------
Installing package
'/var/tmp/jinstall-14.1-20140114_ib_14_1_psd.1-domestic-signed.tgz' ...
Verified jinstall-14.1-20140114_ib_14_1_psd.1-domestic.tgz signed by
PackageDevelopmentEc_2014
Adding jinstall...

WARNING:     The software that is being installed has limited support.
WARNING:     Run 'file show /etc/notices/unsupported.txt' for details.

veriexec: accepting signer: PackageDevelopmentEc_2014
Verified manifest signed by PackageDevelopmentEc_2014

WARNING:     This package will load JUNOS 14.1-20140114_ib_14_1_psd.1 software.
WARNING:     It will save JUNOS configuration files, and SSH keys
```

```
WARNING:     (if configured), but erase all other files and information
WARNING:     stored on this machine.  It will attempt to preserve dumps
WARNING:     and log files, but this can not be guaranteed.  This is the
WARNING:     pre-installation stage and all the software is loaded when
WARNING:     you reboot the system.

Saving the config files ...
NOTICE: uncommitted changes have been saved in
/var/db/config/juniper.conf.pre-install
Installing the bootstrap installer ...

WARNING:     A REBOOT IS REQUIRED TO LOAD THIS SOFTWARE CORRECTLY. Use the
WARNING:     'request system reboot' command when software installation is
WARNING:     complete. To abort the installation, do not reboot your system,
WARNING:     instead use the 'request system software delete jinstall'
WARNING:     command as soon as this operation completes.

Saving package file in
/var/sw/pkg/jinstall-14.1-20140114_ib_14_1_psd.1-domestic-signed.tgz ...
Saving state for rollback ...

member1-re1:
--------------------------------------------------------------------------
Installing package
'/var/tmp/jinstall-14.1-20140114_ib_14_1_psd.1-domestic-signed.tgz' ...
Verified jinstall-14.1-20140114_ib_14_1_psd.1-domestic.tgz signed by
PackageDevelopmentEc_2014
Adding jinstall...

WARNING:     The software that is being installed has limited support.
WARNING:     Run 'file show /etc/notices/unsupported.txt' for details.

veriexec: accepting signer: PackageDevelopmentEc_2014
Verified manifest signed by PackageDevelopmentEc_2014

WARNING:     This package will load JUNOS 14.1-20140114_ib_14_1_psd.1 software.
WARNING:     It will save JUNOS configuration files, and SSH keys
WARNING:     (if configured), but erase all other files and information
WARNING:     stored on this machine.  It will attempt to preserve dumps
WARNING:     and log files, but this can not be guaranteed.  This is the
WARNING:     pre-installation stage and all the software is loaded when
WARNING:     you reboot the system.

Saving the config files ...
NOTICE: uncommitted changes have been saved in
/var/db/config/juniper.conf.pre-install
Installing the bootstrap installer ...

WARNING:     A REBOOT IS REQUIRED TO LOAD THIS SOFTWARE CORRECTLY. Use the
WARNING:     'request system reboot' command when software installation is
WARNING:     complete. To abort the installation, do not reboot your system,
WARNING:     instead use the 'request system software delete jinstall'
WARNING:     command as soon as this operation completes.

Saving package file in
/var/sw/pkg/jinstall-14.1-20140114_ib_14_1_psd.1-domestic-signed.tgz ...
Saving state for rollback ...
Installing package
'/var/tmp/jinstall-14.1-20140114_ib_14_1_psd.1-domestic-signed.tgz' ...
Verified jinstall-14.1-20140114_ib_14_1_psd.1-domestic.tgz signed by
PackageDevelopmentEc_2014
```

```
Adding jinstall...

WARNING:     The software that is being installed has limited support.
WARNING:     Run 'file show /etc/notices/unsupported.txt' for details.

veriexec: accepting signer: PackageDevelopmentEc_2014
Verified manifest signed by PackageDevelopmentEc_2014

WARNING:     This package will load JUNOS 14.1-20140114_ib_14_1_psd.1 software.
WARNING:     It will save JUNOS configuration files, and SSH keys
WARNING:     (if configured), but erase all other files and information
WARNING:     stored on this machine.  It will attempt to preserve dumps
WARNING:     and log files, but this can not be guaranteed.  This is the
WARNING:     pre-installation stage and all the software is loaded when
WARNING:     you reboot the system.

Saving the config files ...
NOTICE: uncommitted changes have been saved in
/var/db/config/juniper.conf.pre-install
Installing the bootstrap installer ...

WARNING:     A REBOOT IS REQUIRED TO LOAD THIS SOFTWARE CORRECTLY. Use the
WARNING:     'request system reboot' command when software installation is
WARNING:     complete. To abort the installation, do not reboot your system,
WARNING:     instead use the 'request system software delete jinstall'
WARNING:     command as soon as this operation completes.

Saving package file in
/var/sw/pkg/jinstall-14.1-20140114_ib_14_1_psd.1-domestic-signed.tgz ...
Saving state for rollback ...
[Jan 30 11:03:12]:ISSU: Backup RE Prepare Done
Rebooting standby RE's
Sending Reboot Command to member0-re0
Shutdown NOW!
Reboot consistency check bypassed - jinstall 14.1-20140114_ib_14_1_psd.1 will
complete installation upon reboot
[pid 2757]
Sending Reboot Command to member1-re1
Shutdown NOW!
Reboot consistency check bypassed - jinstall 14.1-20140114_ib_14_1_psd.1 will
complete installation upon reboot
[pid 2670]
Waiting for standby RE's to boot
[Jan 30 11:18:26]:ISSU: LCC Backup REs Prepare Done
Waiting for standby RE's to have the correct ISSU state
Waiting for protocol backup to be ready to switch mastership
Switching mastership on the protocol backup chassis to slot 1
Waiting for protocol backup chassis master switch to complete
Globally updating ISSU state
Waiting for protocol backup chassis to become GRES ready
[Jan 30 11:19:18]:ISSU: VC Protocol Backup has Switched
Passing ISSU control to chassisd
Chassis ISSU Started
[Jan 30 11:21:01]:ISSU: Preparing Daemons
[Jan 30 11:22:02]:ISSU: Daemons Ready for ISSU
[Jan 30 11:22:06]:ISSU: Starting Upgrade for FRUs
[Jan 30 11:25:42]:ISSU: Preparing for Switchover
[Jan 30 11:26:06]:ISSU: Ready for Switchover
[Jan 30 11:26:20]:ISSU: All VC Members Ready for Switchover
Waiting for master chassis to be switch ready
Switching mastership locally
```

```
Resolving mastership...
Complete. The other routing engine becomes the master.
Waiting for virtual chassis roles to switch
Globally updating ISSU state to IDLE
[Jan 30 11:26:33]:ISSU: IDLE
Rebooting protocol backup standby RE.
Sending Reboot Command to member1-re0

member1-re0:
--------------------------------------------------------------------------
Shutdown NOW!
Reboot consistency check bypassed - jinstall 14.1-20140114_ib_14_1_psd.1 will
complete installation upon reboot
[pid 10462]
Rebooting locally to complete the in service upgrade.
Shutdown NOW!
Reboot consistency check bypassed - jinstall 14.1-20140114_ib_14_1_psd.1 will
complete installation upon reboot
[pid 13458]

{local:member0-re1}
user@host>
*** FINAL System shutdown message from user@host ***

System going down IMMEDIATELY


Connection closed by foreign host.
```

request virtual-chassis member-id delete (MX Series Virtual Chassis)

Syntax request virtual-chassis member-id delete

Release Information Command introduced in Junos OS Release 11.2.
Command introduced in Junos OS Release 13.2R2 for EX Series switches.

Description Remove (**delete**) the member ID from a router or switch that you want to remove from a Virtual Chassis configuration.

NOTE: Issuing the command to remove the member ID causes the device to reboot, and requires you to confirm that you want to proceed with this operation. If you do not confirm the operation, the software cancels the command.

Required Privilege Level system-control

Related Documentation
- Deleting Member IDs in a Virtual Chassis Configuration on page 97
- Example: Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 111

List of Sample Output request virtual-chassis member-id delete on page 271

Sample Output

request virtual-chassis member-id delete

```
user@host> request virtual-chassis member-id delete
This command will disable virtual-chassis mode and reboot the system.

 Continue? [yes,no] (no)
```

request virtual-chassis member-id set

Syntax request virtual-chassis member-id set member *member-id*

Syntax (MX960, MX2010, and MX2020 Routers) request virtual-chassis member-id set member *member-id* <slots-per-chassis *slot-count*>

Release Information Command introduced in Junos OS Release 11.2.
Command introduced in Junos OS Release 13.2R2 for EX Series switches.
slots-per-chassis option added in Junos OS Release 15.1 for MX960 routers, MX2010 routers, and MX2020 routers.

Description Assign (**set**) a member ID and, optionally, a slot count to a router or switch that you want to add as a member of a Virtual Chassis configuration.

NOTE: Issuing the **request virtual-chassis member-id set** command causes the device to reboot, and requires you to confirm that you want to proceed with this operation. If you do not confirm the operation, the software cancels the command. After the reboot all MPCs remain powered off until the Virtual Chassis port connection is configured.

Options

member ***member-id***—Assign the numeric value that identifies a member router or switch in a Virtual Chassis configuration. When you assign a member ID to a router or switch, assign the same member ID defined for this device in the preprovisioned configuration. Replace ***member-id*** with the value 0 or 1.

slots-per-chassis *slot-count*—(MX960, MX2010, and MX2020 routers) (Optional) Identify the number of chassis slots in the Virtual Chassis member router. To ensure that a Virtual Chassis consisting of an MX2020 member router and either an MX960 or MX2010 member router forms properly, you must explicitly set the *slot-count* value for the MX960 router or MX2010 router to 20 to match the slot count of the MX2020 router.

Values:

The valid values for ***slot-count*** are as follows:

- MX960 router: 12 or 20
- MX2010 router: 12 or 20
- MX2020 router: 20

Default:

The default values for ***slot-count*** are as follows:

- MX960 router: 12

- MX2010 router: 12
- MX2020 router: 20

Required Privilege Level system-control

Related Documentation
- Configuring Member IDs for a Virtual Chassis on page 72
- *Configuring an EX9200 Virtual Chassis*
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

List of Sample Output request virtual-chassis member-id set (Assigning a Member ID) on page 273
request virtual-chassis member-id set (Assigning a Member ID and Slot Count) on page 273

Sample Output

request virtual-chassis member-id set (Assigning a Member ID)

```
user@host> request virtual-chassis member-id set member 0
This command will enable virtual-chassis mode and reboot the system.

 Continue? [yes,no] (no)
```

request virtual-chassis member-id set (Assigning a Member ID and Slot Count)

```
user@host> request virtual-chassis member-id set member 1 slots-per-chassis 20
This command will enable virtual-chassis mode and reboot the system.

 Continue? [yes,no] (no)
```

request virtual-chassis routing-engine master switch

Syntax request virtual-chassis routing-engine master switch
<check>

Release Information Command introduced in Junos OS Release 11.2.
Option **check** introduced in Junos OS Release 12.2.
Command introduced in Junos OS Release 13.2R2 for EX Series switches.

Description Change the mastership in an MX Series Virtual Chassis or EX9200 Virtual Chassis by switching the global roles of the master router or switch and backup router or switch in the Virtual Chassis configuration. The **request virtual-chassis routing-engine master switch** command must be issued from the master router or switch (VC-Mm).

(MX Series routers only) The local roles (**master** and **standby**, or **m** and **s**) of the Routing Engines in the Virtual Chassis master router change after a global switchover, but the local roles of the Routing Engines in the Virtual Chassis backup router do not change. For example, the master Routing Engine in the Virtual Chassis master router (VC-Mm) becomes the standby Routing Engine in the Virtual Chassis backup router (VC-Bs) after the global switchover. By contrast, the master Routing Engine in the Virtual Chassis backup router (VC-Bm) remains the master Routing Engine in the Virtual Chassis master router (VC-Mm) after the global switchover.

NOTE: Before you issue the **request virtual-chassis routing-engine master switch** command from the master router or switch in the Virtual Chassis, make sure that the system configuration is synchronized between the master and backup router or switch. If the configuration is not synchronized, or if you attempt to issue the **request virtual-chassis routing-engine master switch** command from the backup router or switch instead of from the master router or switch, the device displays an error message and rejects the command.

If you issue the **request virtual-chassis routing-engine master switch** command when the Virtual Chassis is in a transition state (for example, the backup router or switch is disconnecting from the Virtual Chassis), the device does not process the command.

Options **check**—(Optional) Perform a check from the master router or switch to determine whether the member routers or switches are ready for GRES from a database synchronization perspective, without initiating the GRES operation itself.

Required Privilege Level system-control

Related Documentation
- Switching the Global Master and Backup Roles in a Virtual Chassis Configuration on page 95

- Determining GRES Readiness in a Virtual Chassis Configuration on page 185
- Switchover Behavior in an MX Series Virtual Chassis on page 34
- Mastership Election in a Virtual Chassis on page 32

List of Sample Output

request virtual-chassis routing-engine master switch (From Master Router) on page 275
request virtual-chassis routing-engine master switch (Error When Configuration Not Synchronized) on page 275
request virtual-chassis routing-engine master switch (Error When Run from Backup Router) on page 275
request virtual-chassis routing-engine master switch check (Ready for GRES) on page 275
request virtual-chassis routing-engine master switch check (Not Ready for GRES) on page 275

Sample Output

request virtual-chassis routing-engine master switch (From Master Router)

```
{master:member0-re0}

user@host> request virtual-chassis routing-engine master switch
Do you want to continue ? [yes,no] (no)
```

request virtual-chassis routing-engine master switch (Error When Configuration Not Synchronized)

```
{master:member0-re0}

user@host> request virtual-chassis routing-engine master switch
Error: mastership switch request NOT honored, backup not ready
```

request virtual-chassis routing-engine master switch (Error When Run from Backup Router)

```
{backup:member1-re0}

user@host> request virtual-chassis routing-engine master switch
error: Virtual Chassis member is not the protocol master
```

request virtual-chassis routing-engine master switch check (Ready for GRES)

```
{master:member0-re0}

user@host> request virtual-chassis routing-engine master switch check
Switchover Ready
```

request virtual-chassis routing-engine master switch check (Not Ready for GRES)

```
{master:member0-re0}

user@host> request virtual-chassis routing-engine master switch check
error: chassisd Not ready for mastership switch, try after 217 secs.
mastership switch request NOT honored, backup not ready
```

request virtual-chassis vc-port delete (MX Series Virtual Chassis)

Syntax request virtual-chassis vc-port delete fpc-slot *fpc-slot-number* pic-slot *pic-slot-number* port *port-number*
<(local | member *member-id*)>

Release Information Command introduced in Junos OS Release 11.2.

Description Remove (**delete**) a Virtual Chassis port from a member router in an MX Series Virtual Chassis configuration. After a Virtual Chassis port is created, it is renamed **vcp-*slot/pic/port***, and is no longer available for configuration as a standard network port. After you remove a Virtual Chassis port, it becomes available to the global configuration and can again function as a standard network port.

NOTE: If the member ID has not been set on the router where you issue the **request virtual-chassis vc-port delete** command, the software prevents the removal of the Virtual Chassis port on the router. To set the member ID, use the request virtual-chassis member-id set command.

Options

fpc-slot *fpc-slot-number*—Number of the Flexible PIC Concentrator (FPC) slot on which the Virtual Chassis port resides. The slot number corresponds to the Modular Port Concentrator (MPC) slot number. Replace ***fpc-slot-number*** with a value appropriate for your router:

- MX960 router—0 through 11.
- MX480 router—0 through 5.
- MX240 router—0 through 2.

pic-slot *pic-slot-number*—Number of the PIC slot on which the Virtual Chassis port resides. Replace ***pic-slot-number*** with a value in the range 0 through 3.

port *port-number*—Number of the port on the PIC on which the Virtual Chassis port resides. Replace ***port-number*** with a value appropriate for your PIC.

local—(Optional) Delete the Virtual Chassis port on the member router on which you are issuing the command. This is the default behavior if you do not specify the **local** or **member** options.

member *member-id*—(Optional) Numeric value that identifies the remote Virtual Chassis member on which you want to delete the Virtual Chassis port. Replace ***member-id*** with the value 0 or 1.

Required Privilege Level system-control

Related Documentation

- Deleting Virtual Chassis Ports in a Virtual Chassis Configuration on page 138
- Example: Deleting a Virtual Chassis Configuration for MX Series 3D Universal Edge Routers on page 111

List of Sample Output request virtual-chassis vc-port delete (Remove vcp-3/2/1) on page 277

Sample Output

request virtual-chassis vc-port delete (Remove vcp-3/2/1)

```
user@host> request virtual-chassis vc-port delete fpc-slot 3 pic-slot 2 port 1
vc-port successfully deleted
```

request virtual-chassis vc-port set (MX Series Virtual Chassis)

Syntax request virtual-chassis vc-port set fpc-slot *fpc-slot-number* pic-slot *pic-slot-number* port *port-number*
<(local | member *member-id*)>

Release Information Command introduced in Junos OS Release 11.2.

Description Create (**set**) a Virtual Chassis port on an MX Series router through which the router connects to other member routers in the Virtual Chassis. You can create Virtual Chassis ports only on Modular Port Concentrator/Modular Interface Card (MPC/MIC) network ports on MX Series routers.

After a Virtual Chassis port is created, it is renamed **vcp-*slot/pic/port***, and is no longer available for configuration as a standard network port. Virtual Chassis ports can be used only to interconnect the MX Series routers in the Virtual Chassis.

NOTE: If the member ID has not been set on the router where you issue the **request virtual-chassis vc-port set** command, the software prevents the creation of the Virtual Chassis port on the router. To set the member ID, use the request virtual-chassis member-id set command.

Options **fpc-slot *fpc-slot-number***—Number of the Flexible PIC Concentrator (FPC) slot on which the Virtual Chassis port resides. The slot number corresponds to the Modular Port Concentrator (MPC) slot number. Replace ***fpc-slot-number*** with a value appropriate for your router:

- MX960 router—0 through 11.
- MX480 router—0 through 5.
- MX240 router—0 through 2.

When you issue the **show interfaces** command on a member router in an MX Series Virtual Chassis, the FPC slot number displayed in the command output reflects the FPC slot numbering and offset used in the Virtual Chassis instead of the physical slot number where the FPC is actually installed. The router with member ID 0 in the Virtual Chassis uses FPC slot numbers 0 through 11 with no offset, and the router with member ID 1 uses FPC slot numbers 12 through 23, with an offset of 12. For example, a 10-Gigabit Ethernet interface that appears as **xe-14/2/2** (FPC slot 14, PIC slot 2, port 2) in the **show interfaces** command is actually interface **xe-2/2/2** (FPC slot 2, PIC slot 2, port 2) on member ID 1 after deducting the FPC slot numbering offset of 12 for member ID 1.

pic-slot *pic-slot-number*—Number of the PIC slot on which the Virtual Chassis port resides. Replace ***pic-slot-number*** with a value in the range 0 through 3.

port ***port-number***—Number of the port on the PIC on which the Virtual Chassis port resides. Replace ***port-number*** with a value appropriate for your PIC.

local—(Optional) Set the Virtual Chassis port on the member router on which you are issuing the command. This is the default behavior if you do not specify the **local** or **member** options.

member ***member-id***—(Optional) Numeric value that identifies the remote Virtual Chassis member on which you want to create the Virtual Chassis port. Replace ***member-id*** with the value 0 or 1.

Required Privilege Level system-control

Related Documentation

- Configuring Virtual Chassis Ports to Interconnect Member Routers or Switches on page 136
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- Guidelines for Configuring Virtual Chassis Ports on page 134

List of Sample Output

request virtual-chassis vc-port set (No Existing Network Port) on page 279
request virtual-chassis vc-port set (Existing Network Port Converted) on page 279
request virtual-chassis vc-port set (On Local Router) on page 279
request virtual-chassis vc-port set (On Remote Member Router 1) on page 279

Sample Output

request virtual-chassis vc-port set (No Existing Network Port)

```
user@host> request virtual-chassis vc-port set fpc-slot 1 pic-slot 1 port 0
vc-port successfully set
```

request virtual-chassis vc-port set (Existing Network Port Converted)

```
user@host> request virtual-chassis vc-port set fpc-slot 2 pic-slot 1 port 1
Port conversion initiated, use “show virtual chassis vc-port” to verify
```

request virtual-chassis vc-port set (On Local Router)

```
user@host> request virtual-chassis vc-port set fpc-slot 2 pic-slot 1 port 3 local
vc-port successfully set
```

request virtual-chassis vc-port set (On Remote Member Router 1)

```
user@host> request virtual-chassis vc-port set fpc-slot 5 pic-slot 3 port 10 member 1
vc-port successfully set
```

show chassis network-services

Syntax show chassis network-services

Release Information
Command introduced in Junos OS Release 9.4.
Command introduced in Junos OS Release 12.3 for MX2010 3D Universal Edge Routers.
Command introduced in Junos OS Release 12.3 for MX2020 3D Universal Edge Routers.
Command introduced in Junos OS Release 13.2 for MX104 3D Universal Edge Routers.
Command introduced in Junos OS Release 15.F5 for PTX Series Routers with third-generation FPCs.
Command introduced in Junos OS Release 17.2 for PTX10008 Routers.
Command introduced in Junos OS Release 17.2 for MX2008 3D Universal Edge Routers.

Description Display the network services mode that the router is configured to run in—IP Network Services mode, Ethernet Network Services mode, Enhanced IP Network Services mode, Enhanced Ethernet Network Services mode, or Enhanced mode.

Options This command has no options.

Required Privilege Level view

Related Documentation
- *enhanced-mode*

List of Sample Output
show chassis network services on page 281
show chassis network services (MX104 Router) on page 281
show chassis network services (MX2010 Router) on page 281
show chassis network services (MX2020 Router) on page 281
show chassis network services (MX2008 Router) on page 281
show chassis network services (PTX Router with third-generation FPCs) on page 281

Output Fields Table 28 on page 280 lists the output fields for the **show chassis network services** command. Output fields are listed in the approximate order in which they appear.

Table 28: show chassis network services Output Fields

Field Name	Field Description
Network Services Mode	Network services mode configured for the router: • **IP**—IP Network Services mode. • **Ethernet**—Ethernet Network Services mode. • **enhanced-ip**—Enhanced IP Network Services mode • **enhanced-ethernet**—Enhanced Ethernet Network Services mode • **Enhanced-Mode**—Enhanced mode for PTX Series routers that have third-generation FPCs installed. See *enhanced-mode*.

Sample Output

show chassis network services

```
user@host> show chassis network services
Network Services Mode: IP
```

show chassis network services (MX104 Router)

```
user@host> show chassis network services
Network Services Mode: IP
```

show chassis network services (MX2010 Router)

```
user@host> show chassis network services
Network Services Mode: Enhanced-IP
```

show chassis network services (MX2020 Router)

```
user@host> show chassis network services
Network Services Mode: Enhanced-IP
```

show chassis network services (MX2008 Router)

```
user@host> show chassis network-services
Network Services Mode: Enhanced-IP
```

show chassis network services (PTX Router with third-generation FPCs)

```
user@host> show chassis network services
Network Services Mode: Enhanced-Mode
```

show interfaces vcp

Syntax

show interfaces vcp-*fpc/pic/port*
<brief | detail | extensive | terse>
<descriptions>
<media>
<snmp-index *snmp-index*>
<statistics>

Release Information

Command introduced in Junos OS Release 11.4.

Description

Display status information about the specified Virtual Chassis Port.

NOTE: This command only displays the statistics for the local interface of the router that you are directly logged in to. You cannot display interface status information of other member routers.

Options

vcp-*fpc/pic/port* —Display standard information about the specified Virtual Chassis Port.

brief | detail | extensive | terse—(Optional) Display the specified level of output.

descriptions—(Optional) Display interface description strings.

media—(Optional) Display media-specific information about network interfaces.

snmp-index *snmp-index*—(Optional) Display information for the specified SNMP index of the interface.

statistics—(Optional) Display static interface statistics.

Additional Information

In a logical system, this command displays information only about the logical interfaces and not about the physical interfaces.

Required Privilege Level

view

Related Documentation

- Guidelines for Configuring Virtual Chassis Ports on page 134
- show virtual-chassis protocol interface (MX Series Virtual Chassis) on page 321

List of Sample Output

show interfaces vcp (specific interface) on page 297
show interfaces vcp brief (specific interface) on page 298
show interfaces vcp detail (specific interface) on page 298

Table 29: show interfaces vcp Output Fields

Output Fields

Field Name	Field Description	Level of Output
Physical Interface		
Physical Interface	Name of the physical interface.	All levels
Enabled	State of the interface. Possible values are described in the "Enabled Field" section under *Common Output Fields Description*.	All levels
Interface index	Index number of the physical interface, which reflects its initialization sequence.	**detail extensive** none
SNMP ifIndex	SNMP index number for the physical interface.	**detail extensive** none
Generation	Unique number for use by Juniper Networks technical support only.	**detail extensive**
Link-level type	Encapsulation being used on the physical interface.	All levels
MTU	Maximum transmission unit size on the physical interface.	All levels
Speed	Speed at which the interface is running.	All levels
Loopback	Loopback status: **Enabled** or **Disabled**. If loopback is enabled, type of loopback: **Local** or **Remote**.	All levels
Source filtering	Source filtering status: **Enabled** or **Disabled**.	All levels
LAN-PHY mode	10-Gigabit Ethernet interface operating in Local Area Network Physical Layer Device (LAN PHY) mode. LAN PHY allows 10-Gigabit Ethernet wide area links to use existing Ethernet applications.	All levels
WAN-PHY mode	10-Gigabit Ethernet interface operating in Wide Area Network Physical Layer Device (WAN PHY) mode. WAN PHY allows 10-Gigabit Ethernet wide area links to use fiber-optic cables and other devices intended for SONET/SDH.	All levels
Unidirectional	Unidirectional link mode status for 10-Gigabit Ethernet interface: **Enabled** or **Disabled** for parent interface; **Rx-only** or **Tx-only** for child interfaces.	All levels
Flow control	Flow control status: **Enabled** or **Disabled**.	All levels
Auto-negotiation	(Gigabit Ethernet interfaces) Autonegotiation status: **Enabled** or **Disabled**.	All levels
Remote-fault	(Gigabit Ethernet interfaces) Remote fault status: • **Online**—Autonegotiation is manually configured as online. • **Offline**—Autonegotiation is manually configured as offline.	All levels
Device flags	Information about the physical device. Possible values are described in the "Device Flags" section under *Common Output Fields Description*.	All levels
Interface flags	Information about the interface. Possible values are described in the "Interface Flags" section under *Common Output Fields Description*.	All levels

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
Link flags	Information about the link. Possible values are described in the "Links Flags" section under *Common Output Fields Description*.	All levels
Wavelength	(10-Gigabit Ethernet dense wavelength-division multiplexing [DWDM] interfaces) Displays the configured wavelength, in nanometers (nm).	All levels
Frequency	(10-Gigabit Ethernet DWDM interfaces only) Displays the frequency associated with the configured wavelength, in terahertz (THz).	All levels
CoS queues	Number of CoS queues configured.	**detail extensive** none
Schedulers	(Gigabit Ethernet intelligent queuing 2 [IQ2] interfaces only) Number of CoS schedulers configured.	**extensive**
Hold-times	Current interface hold-time up and hold-time down, in milliseconds (ms).	**detail extensive**
Current address	Configured MAC address.	**detail extensive** none
Hardware address	Hardware MAC address.	**detail extensive** none
Last flapped	Date, time, and how long ago the interface went from down to up. The format is **Last flapped:** ***year-month-day hour:minute:second:timezone*** **(*hour:minute:second* ago)**. For example, **Last flapped: 2002-04-26 10:52:40 PDT (04:33:20 ago).**	**detail extensive** none
Input Rate	Input rate in bits per second (bps) and packets per second (pps). The value in this field also includes the Layer 2 overhead bytes for ingress traffic on Ethernet interfaces if you enable accounting of Layer 2 overhead at the PIC level or the logical interface level.	None
Output Rate	Output rate in bps and pps. The value in this field also includes the Layer 2 overhead bytes for egress traffic on Ethernet interfaces if you enable accounting of Layer 2 overhead at the PIC level or the logical interface level.	None
Statistics last cleared	Time when the statistics for the interface were last set to zero.	**detail extensive**
Egress account overhead	Layer 2 overhead in bytes that is accounted in the interface statistics for egress traffic.	**detail extensive**
Ingress account overhead	Layer 2 overhead in bytes that is accounted in the interface statistics for ingress traffic.	**detail extensive**

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
Traffic statistics	Number and rate of bytes and packets received and transmitted on the physical interface. • **Input bytes**—Number of bytes received on the interface. The value in this field also includes the Layer 2 overhead bytes for ingress traffic on Ethernet interfaces if you enable accounting of Layer 2 overhead at the PIC level or the logical interface level. • **Output bytes**—Number of bytes transmitted on the interface. The value in this field also includes the Layer 2 overhead bytes for egress traffic on Ethernet interfaces if you enable accounting of Layer 2 overhead at the PIC level or the logical interface level. • **Input packets**—Number of packets received on the interface. • **Output packets**—Number of packets transmitted on the interface. Gigabit Ethernet and 10-Gigabit Ethernet IQ PICs count the overhead and CRC bytes. For Gigabit Ethernet IQ PICs, the input byte counts vary by interface type. For more information, see Table 31 under the *show interfaces (10-Gigabit Ethernet)* command.	**detail extensive**
Input errors	Input errors on the interface. The following paragraphs explain the counters whose meaning might not be obvious: • **Errors**—Sum of the incoming frame aborts and FCS errors. • **Drops**—Number of packets dropped by the input queue of the I/O Manager ASIC. If the interface is saturated, this number increments once for every packet that is dropped by the ASIC's RED mechanism. • **Framing errors**—Number of packets received with an invalid frame checksum (FCS). • **Runts**—Number of frames received that are smaller than the runt threshold. • **Policed discards**—Number of frames that the incoming packet match code discarded because they were not recognized or not of interest. Usually, this field reports protocols that Junos OS does not handle. • **L3 incompletes**—Number of incoming packets discarded because they failed Layer 3 (usually IPv4) sanity checks of the header. For example, a frame with less than 20 bytes of available IP header is discarded. L3 incomplete errors can be ignored by configuring the **ignore-l3-incompletes** statement. • **L2 channel errors**—Number of times the software did not find a valid logical interface for an incoming frame. • **L2 mismatch timeouts**—Number of malformed or short packets that caused the incoming packet handler to discard the frame as unreadable. • **FIFO errors**—Number of FIFO errors in the receive direction that are reported by the ASIC on the PIC. If this value is ever nonzero, the PIC is probably malfunctioning. • **Resource errors**—Sum of transmit drops.	**extensive**

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
Output errors	Output errors on the interface. The following paragraphs explain the counters whose meaning might not be obvious: • **Carrier transitions**—Number of times the interface has gone from **down** to **up**. This number does not normally increment quickly, increasing only when the cable is unplugged, the far-end system is powered down and then up, or another problem occurs. If the number of carrier transitions increments quickly (perhaps once every 10 seconds), the cable, the far-end system, or the PIC or PIM is malfunctioning. • **Errors**—Sum of the outgoing frame aborts and FCS errors. • **Drops**—Number of packets dropped by the output queue of the I/O Manager ASIC. If the interface is saturated, this number increments once for every packet that is dropped by the ASIC's RED mechanism. NOTE: Due to accounting space limitations on certain Type 3 FPCs (which are supported in M320 and T640 routers), the **Drops** field does not always use the correct value for queue 6 or queue 7 for interfaces on 10-port 1-Gigabit Ethernet PICs. • **Collisions**—Number of Ethernet collisions. The Gigabit Ethernet PIC supports only full-duplex operation, so for Gigabit Ethernet PICs, this number should always remain 0. If it is nonzero, there is a software bug. • **Aged packets**—Number of packets that remained in shared packet SDRAM so long that the system automatically purged them. The value in this field should never increment. If it does, it is most likely a software bug or possibly malfunctioning hardware. • **FIFO errors**—Number of FIFO errors in the send direction as reported by the ASIC on the PIC. If this value is ever nonzero, the PIC is probably malfunctioning. • **HS link CRC errors**—Number of errors on the high-speed links between the ASICs responsible for handling the router interfaces. • **MTU errors**—Number of packets whose size exceeded the MTU of the interface. • **Resource errors**—Sum of transmit drops.	**extensive**
Egress queues	Total number of egress queues supported on the specified interface. NOTE: In DPCs that are not of the enhanced type, such as DPC 40x 1GE R, DPCE 20x 1GE + 2x 10GE R, or DPCE 40x 1GE R, you might notice a discrepancy in the output of the **show interfaces** command because incoming packets might be counted in the Egress queues section of the output. This problem occurs on non-enhanced DPCs because the egress queue statistics are polled from IMQ (Inbound Message Queuing) block of the I-chip. The IMQ block does not differentiate between ingress and egress WAN traffic; as a result, the combined statistics are displayed in the egress queue counters on the Routing Engine. In a simple VPLS scenorio, if there is no MAC entry in DMAC table (by sending unidirectional traffic), traffic is flooded and the input traffic is accounted in IMQ. For bidirectional traffic (MAC entry in DMAC table), if the outgoing interface is on the same I-chip then both ingress and egress statistics are counted in a combined way. If the outgoing interface is on a different I-chip or FPC, then only egress statistics are accounted in IMQ. This behavior is expected with non-enhanced DPCs	**detail extensive**

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
Queue counters (Egress)	CoS queue number and its associated user-configured forwarding class name. • **Queued packets**—Number of queued packets. • **Transmitted packets**—Number of transmitted packets. • **Dropped packets**—Number of packets dropped by the ASIC's RED mechanism. NOTE: Due to accounting space limitations on certain Type 3 FPCs (which are supported in M320 and T640 routers), the **Dropped packets** field does not always display the correct value for queue 6 or queue 7 for interfaces on 10-port 1-Gigabit Ethernet PICs.	**detail extensive**
Ingress queues	Total number of ingress queues supported on the specified interface. Displayed on IQ2 interfaces.	**extensive**
Queue counters (Ingress)	CoS queue number and its associated user-configured forwarding class name. Displayed on IQ2 interfaces. • **Queued packets**—Number of queued packets. • **Transmitted packets**—Number of transmitted packets. • **Dropped packets**—Number of packets dropped by the ASIC's RED mechanism.	**extensive**
Active alarms and Active defects	Ethernet-specific defects that can prevent the interface from passing packets. When a defect persists for a certain amount of time, it is promoted to an alarm. Based on the router configuration, an alarm can ring the red or yellow alarm bell on the router, or turn on the red or yellow alarm LED on the craft interface. These fields can contain the value **None** or **Link**. • **None**—There are no active defects or alarms. • **Link**—Interface has lost its link state, which usually means that the cable is unplugged, the far-end system has been turned off, or the PIC is malfunctioning.	**detail extensive** none
Interface transmit statistics	(On MX Series devices) Status of the **interface-transmit-statistics** configuration: Enabled or Disabled. • **Enabled**—When the **interface-transmit-statistics** statement is included in the configuration. If this is configured, the interface statistics show the actual transmitted load on the interface. • **Disabled**—When the **interface-transmit-statistics** statement is not included in the configuration. If this is not configured, the interface statistics show the offered load on the interface.	**detail extensive**
OTN FEC statistics	The forward error correction (FEC) counters provide the following statistics: • **Corrected Errors**—The count of corrected errors in the last second. • **Corrected Error Ratio**—The corrected error ratio in the last 25 seconds. For example, 1e-7 is 1 error per 10 million bits.	**detail extensive**

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
PCS statistics	(10-Gigabit Ethernet interfaces) Displays Physical Coding Sublayer (PCS) fault conditions from the WAN PHY or the LAN PHY device. • **Bit errors**—The number of seconds during which at least one bit error rate (BER) occurred while the PCS receiver is operating in normal mode. • **Errored blocks**—The number of seconds when at least one errored block occurred while the PCS receiver is operating in normal mode.	**detail extensive**

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
MAC statistics	**Receive** and **Transmit** statistics reported by the PIC's MAC subsystem, including the following: • **Total octets** and **total packets**—Total number of octets and packets. For Gigabit Ethernet IQ PICs, the received octets count varies by interface type. For more information, see Table 31 under the *show interfaces (10-Gigabit Ethernet)* command. • **Unicast packets, Broadcast packets,** and **Multicast packets**—Number of unicast, broadcast, and multicast packets. • **CRC/Align errors**—Total number of packets received that had a length (excluding framing bits, but including FCS octets) of between 64 and 1518 octets, inclusive, and had either a bad FCS with an integral number of octets (FCS Error) or a bad FCS with a nonintegral number of octets (Alignment Error). • **FIFO error**—Number of FIFO errors that are reported by the ASIC on the PIC. If this value is ever nonzero, the PIC or a cable is probably malfunctioning. • **MAC control frames**—Number of MAC control frames. • **MAC pause frames**—Number of MAC control frames with **pause** operational code. • **Oversized frames**—There are two possible conditions regarding the number of oversized frames: ◦ Packet length exceeds 1518 octets, or ◦ Packet length exceeds MRU • **Jabber frames**—Number of frames that were longer than 1518 octets (excluding framing bits, but including FCS octets), and had either an FCS error or an alignment error. This definition of jabber is different from the definition in IEEE-802.3 section 8.2.1.5 (10BASE5) and section 10.3.1.4 (10BASE2). These documents define jabber as the condition in which any packet exceeds 20 ms. The allowed range to detect jabber is from 20 ms to 150 ms. • **Fragment frames**—Total number of packets that were less than 64 octets in length (excluding framing bits, but including FCS octets) and had either an FCS error or an alignment error. Fragment frames normally increment because both runts (which are normal occurrences caused by collisions) and noise hits are counted. • **VLAN tagged frames**—Number of frames that are VLAN tagged. The system uses the TPID of 0x8100 in the frame to determine whether a frame is tagged or not. NOTE: The 20-port Gigabit Ethernet MIC (MIC-3D-20GE-SFP) does not have hardware counters for VLAN frames. Therefore, the **VLAN tagged frames** field displays 0 when the **show interfaces** command is executed on a 20-port Gigabit Ethernet MIC. In other words, the number of VLAN tagged frames cannot be determined for the 20-port Gigabit Ethernet MIC. • **Code violations**—Number of times an event caused the PHY to indicate “Data reception error” or “invalid data symbol error.”	**extensive**
OTN Received Overhead Bytes	APS/PCC0: 0x02, APS/PCC1: 0x11, APS/PCC2: 0x47, APS/PCC3: 0x58 Payload Type: 0x08	**extensive**
OTN Transmitted Overhead Bytes	APS/PCC0: 0x00, APS/PCC1: 0x00, APS/PCC2: 0x00, APS/PCC3: 0x00 Payload Type: 0x08	**extensive**

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
Filter statistics	**Receive** and **Transmit** statistics reported by the PIC's MAC address filter subsystem. The filtering is done by the content-addressable memory (CAM) on the PIC. The filter examines a packet's source and destination MAC addresses to determine whether the packet should enter the system or be rejected. • **Input packet count**—Number of packets received from the MAC hardware that the filter processed. • **Input packet rejects**—Number of packets that the filter rejected because of either the source MAC address or the destination MAC address. • **Input DA rejects**—Number of packets that the filter rejected because the destination MAC address of the packet is not on the accept list. It is normal for this value to increment. When it increments very quickly and no traffic is entering the router from the far-end system, either there is a bad ARP entry on the far-end system, or multicast routing is not on and the far-end system is sending many multicast packets to the local router (which the router is rejecting). • **Input SA rejects**—Number of packets that the filter rejected because the source MAC address of the packet is not on the accept list. The value in this field should increment only if source MAC address filtering has been enabled. If filtering is enabled, if the value increments quickly, and if the system is not receiving traffic that it should from the far-end system, it means that the user-configured source MAC addresses for this interface are incorrect. • **Output packet count**—Number of packets that the filter has given to the MAC hardware. • **Output packet pad count**—Number of packets the filter padded to the minimum Ethernet size (60 bytes) before giving the packet to the MAC hardware. Usually, padding is done only on small ARP packets, but some very small IP packets can also require padding. If this value increments rapidly, either the system is trying to find an ARP entry for a far-end system that does not exist or it is misconfigured. • **Output packet error count**—Number of packets with an indicated error that the filter was given to transmit. These packets are usually aged packets or are the result of a bandwidth problem on the FPC hardware. On a normal system, the value of this field should not increment. • **CAM destination filters, CAM source filters**—Number of entries in the CAM dedicated to destination and source MAC address filters. There can only be up to 64 source entries. If source filtering is disabled, which is the default, the values for these fields should be 0.	**extensive**
PMA PHY	(10-Gigabit Ethernet interfaces, WAN PHY mode) SONET error information: • **Seconds**—Number of seconds the defect has been active. • **Count**—Number of times that the defect has gone from inactive to active. • **State**—State of the error. Any state other than **OK** indicates a problem. Subfields are: • **PHY Lock**—Phase-locked loop • **PHY Light**—Loss of optical signal	**extensive**

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
WIS section	(10-Gigabit Ethernet interfaces, WAN PHY mode) SONET error information: • **Seconds**—Number of seconds the defect has been active. • **Count**—Number of times that the defect has gone from inactive to active. • **State**—State of the error. Any state other than **OK** indicates a problem. Subfields are: • **BIP-B1**—Bit interleaved parity for SONET section overhead • **SEF**—Severely errored framing • **LOL**—Loss of light • **LOF**—Loss of frame • **ES-S**—Errored seconds (section) • **SES-S**—Severely errored seconds (section) • **SEFS-S**—Severely errored framing seconds (section)	**extensive**
WIS line	(10-Gigabit Ethernet interfaces, WAN PHY mode) Active alarms and defects, plus counts of specific SONET errors with detailed information: • **Seconds**—Number of seconds the defect has been active. • **Count**—Number of times that the defect has gone from inactive to active. • **State**—State of the error. Any state other than **OK** indicates a problem. Subfields are: • **BIP-B2**—Bit interleaved parity for SONET line overhead • **REI-L**—Remote error indication (near-end line) • **RDI-L**—Remote defect indication (near-end line) • **AIS-L**—Alarm indication signal (near-end line) • **BERR-SF**—Bit error rate fault (signal failure) • **BERR-SD**—Bit error rate defect (signal degradation) • **ES-L**—Errored seconds (near-end line) • **SES-L**—Severely errored seconds (near-end line) • **UAS-L**—Unavailable seconds (near-end line) • **ES-LFE**—Errored seconds (far-end line) • **SES-LFE**—Severely errored seconds (far-end line) • **UAS-LFE**—Unavailable seconds (far-end line)	**extensive**

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
WIS path	(10-Gigabit Ethernet interfaces, WAN PHY mode) Active alarms and defects, plus counts of specific SONET errors with detailed information: • **Seconds**—Number of seconds the defect has been active. • **Count**—Number of times that the defect has gone from inactive to active. • **State**—State of the error. Any state other than **OK** indicates a problem. Subfields are: • **BIP-B3**—Bit interleaved parity for SONET section overhead • **REI-P**—Remote error indication • **LOP-P**—Loss of pointer (path) • **AIS-P**—Path alarm indication signal • **RDI-P**—Path remote defect indication • **UNEQ-P**—Path unequipped • **PLM-P**—Path payload (signal) label mismatch • **ES-P**—Errored seconds (near-end STS path) • **SES-P**—Severely errored seconds (near-end STS path) • **UAS-P**—Unavailable seconds (near-end STS path) • **SES-PFE**—Severely errored seconds (far-end STS path) • **UAS-PFE**—Unavailable seconds (far-end STS path)	**extensive**

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output

Autonegotiation information	Information about link autonegotiation.
	• **Negotiation status:**
	  • **Incomplete**—Ethernet interface has the speed or link mode configured.
	  • **No autonegotiation**—Remote Ethernet interface has the speed or link mode configured, or does not perform autonegotiation.
	  • **Complete**—Ethernet interface is connected to a device that performs autonegotiation and the autonegotiation process is successful.
	• **Link partner status**—**OK** when Ethernet interface is connected to a device that performs autonegotiation and the autonegotiation process is successful.
	• **Link partner**—Information from the remote Ethernet device:
	  • **Link mode**—Depending on the capability of the link partner, either **Full-duplex** or **Half-duplex**.
	  • **Flow control**—Types of flow control supported by the link partner. For Gigabit Ethernet interfaces, types are **Symmetric** (link partner supports **PAUSE** on receive and transmit), **Asymmetric** (link partner supports **PAUSE** on transmit), **Symmetric/Asymmetric** (link partner supports **PAUSE** on receive and transmit or only **PAUSE** on transmit), and **None** (link partner does not support flow control).
	  • **Remote fault**—Remote fault information from the link partner—**Failure** indicates a receive link error. **OK** indicates that the link partner is receiving. **Negotiation error** indicates a negotiation error. **Offline** indicates that the link partner is going offline.
	• **Local resolution**—Information from the local Ethernet device:
	  • **Flow control**—Types of flow control supported by the local device. For Gigabit Ethernet interfaces, advertised capabilities are **Symmetric/Asymmetric** (local device supports **PAUSE** on receive and transmit or only **PAUSE** on receive) and **None** (local device does not support flow control). Depending on the result of the negotiation with the link partner, local resolution flow control type will display **Symmetric** (local device supports **PAUSE** on receive and transmit), **Asymmetric** (local device supports **PAUSE** on receive), and **None** (local device does not support flow control).
	  • **Remote fault**—Remote fault information. **Link OK** (no error detected on receive), **Offline** (local interface is offline), and **Link Failure** (link error detected on receive).
		extensive

Received path trace, Transmitted path trace	(10-Gigabit Ethernet interfaces, WAN PHY mode) SONET/SDH interfaces allow path trace bytes to be sent inband across the SONET/SDH link. Juniper Networks and other router manufacturers use these bytes to help diagnose misconfigurations and network errors by setting the transmitted path trace message so that it contains the system hostname and name of the physical interface. The received path trace value is the message received from the router at the other end of the fiber. The transmitted path trace value is the message that this router transmits.	**extensive**
Packet Forwarding Engine configuration	Information about the configuration of the Packet Forwarding Engine: • **Destination slot**—FPC slot number.	**extensive**

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
CoS information	Information about the CoS queue for the physical interface. • **CoS transmit queue**—Queue number and its associated user-configured forwarding class name. • **Bandwidth %**—Percentage of bandwidth allocated to the queue. • **Bandwidth bps**—Bandwidth allocated to the queue (in bps). • **Buffer %**—Percentage of buffer space allocated to the queue. • **Buffer usec**—Amount of buffer space allocated to the queue, in microseconds. This value is nonzero only if the buffer size is configured in terms of time. • **Priority**—Queue priority: **low** or **high**. • **Limit**—Displayed if rate limiting is configured for the queue. Possible values are **none** and **exact**. If **exact** is configured, the queue transmits only up to the configured bandwidth, even if excess bandwidth is available. If **none** is configured, the queue transmits beyond the configured bandwidth if bandwidth is available.	**extensive**
Logical Interface		
Logical interface	Name of the logical interface.	All levels
Index	Index number of the logical interface, which reflects its initialization sequence.	**detail extensive** none
SNMP ifIndex	SNMP interface index number for the logical interface.	**detail extensive** none
Generation	Unique number for use by Juniper Networks technical support only.	**detail extensive**
Flags	Information about the logical interface. Possible values are described in the "Logical Interface Flags" section under *Common Output Fields Description*.	All levels

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
VLAN-Tag	Rewrite profile applied to incoming or outgoing frames on the outer (**Out**) VLAN tag or for both the outer and inner (**In**) VLAN tags. • **push**—An outer VLAN tag is pushed in front of the existing VLAN tag. • **pop**—The outer VLAN tag of the incoming frame is removed. • **swap**—The outer VLAN tag of the incoming frame is overwritten with the user-specified VLAN tag information. • **push**—An outer VLAN tag is pushed in front of the existing VLAN tag. • **push-push**—Two VLAN tags are pushed in from the incoming frame. • **swap-push**—The outer VLAN tag of the incoming frame is replaced by a user-specified VLAN tag value. A user-specified outer VLAN tag is pushed in front. The outer tag becomes an inner tag in the final frame. • **swap-swap**—Both the inner and the outer VLAN tags of the incoming frame are replaced by the user-specified VLAN tag value. • **pop-swap**—The outer VLAN tag of the incoming frame is removed, and the inner VLAN tag of the incoming frame is replaced by the user-specified VLAN tag value. The inner tag becomes the outer tag in the final frame. • **pop-pop**—Both the outer and inner VLAN tags of the incoming frame are removed.	**brief detail extensive** none
Demux	IP demultiplexing (demux) value that appears if this interface is used as the demux underlying interface. The output is one of the following: • Source Family Inet • Destination Family Inet	**detail extensive** none
Encapsulation	Encapsulation on the logical interface.	All levels
ACI VLAN: Dynamic Profile	Name of the dynamic profile that defines the agent circuit identifier (ACI) interface set. If configured, the ACI interface set enables the underlying Ethernet interface to create dynamic VLAN subscriber interfaces based on ACI information.	**brief detail extensive** none
Protocol	Protocol family. Possible values are described in the "Protocol Field" section under *Common Output Fields Description*.	**detail extensive** none
MTU	Maximum transmission unit size on the logical interface.	**detail extensive** none
Dynamic Profile	(MX Series routers with Trio MPCs only) Name of the dynamic profile that was used to create this interface configured with a Point-to-Point Protocol over Ethernet (PPPoE) family.	**detail extensive** none
Service Name Table	(MX Series routers with Trio MPCs only) Name of the service name table for the interface configured with a PPPoE family.	**detail extensive** none
Max Sessions	(MX Series routers with Trio MPCs only) Maximum number of PPPoE logical interfaces that can be activated on the underlying interface.	**detail extensive** none

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
Duplicate Protection	(MX Series routers with Trio MPCs only) State of PPPoE duplicate protection: **On** or **Off**. When duplicate protection is configured for the underlying interface, a dynamic PPPoE logical interface cannot be activated when an existing active logical interface is present for the same PPPoE client.	**detail extensive** none
Direct Connect	State of the configuration to ignore DSL Forum VSAs: **On** or **Off**. When configured, the router ignores any of these VSAs received from a directly connected CPE device on the interface.	**detail extensive** none
AC Name	Name of the access concentrator.	**detail extensive** none
Maximum labels	Maximum number of MPLS labels configured for the MPLS protocol family on the logical interface.	**detail extensive** none
Traffic statistics	Number and rate of bytes and packets received and transmitted on the specified interface set. • **Input bytes, Output bytes**—Number of bytes received and transmitted on the interface set. The value in this field also includes the Layer 2 overhead bytes for ingress or egress traffic on Ethernet interfaces if you enable accounting of Layer 2 overhead at the PIC level or the logical interface level. • **Input packets, Output packets**—Number of packets received and transmitted on the interface set.	**detail extensive**
IPv6 transit statistics	Number of IPv6 transit bytes and packets received and transmitted on the logical interface if IPv6 statistics tracking is enabled.	**extensive**
Local statistics	Number and rate of bytes and packets destined to the router.	**extensive**
Transit statistics	Number and rate of bytes and packets transiting the switch. NOTE: For Gigabit Ethernet intelligent queuing 2 (IQ2) interfaces, the logical interface egress statistics might not accurately reflect the traffic on the wire when output shaping is applied. Traffic management output shaping might drop packets after they are tallied by the **Output bytes** and **Output packets** interface counters. However, correct values display for both of these egress statistics when per-unit scheduling is enabled for the Gigabit Ethernet IQ2 physical interface, or when a single logical interface is actively using a shared scheduler.	**extensive**
Generation	Unique number for use by Juniper Networks technical support only.	**detail extensive**
Route Table	Route table in which the logical interface address is located. For example, **0** refers to the routing table inet.0.	**detail extensive** none
Flags	Information about protocol family flags. Possible values are described in the "Family Flags" section under *Common Output Fields Description*.	**detail extensive**
Donor interface	(Unnumbered Ethernet) Interface from which an unnumbered Ethernet interface borrows an IPv4 address.	**detail extensive** none

Table 29: show interfaces vcp Output Fields *(continued)*

Field Name	Field Description	Level of Output
Preferred source address	(Unnumbered Ethernet) Secondary IPv4 address of the donor loopback interface that acts as the preferred source address for the unnumbered Ethernet interface.	**detail extensive** none
Input Filters	Names of any input filters applied to this interface. If you specify a precedence value for any filter in a dynamic profile, filter precedence values appear in parentheses next to all interfaces.	**detail extensive**
Output Filters	Names of any output filters applied to this interface. If you specify a precedence value for any filter in a dynamic profile, filter precedence values appear in parentheses next to all interfaces.	**detail extensive**
Mac-Validate Failures	Number of MAC address validation failures for packets and bytes. This field is displayed when MAC address validation is enabled for the logical interface.	**detail extensive** none
Addresses, Flags	Information about the address flags. Possible values are described in the "Addresses Flags" section under *Common Output Fields Description*.	**detail extensive** none
protocol-family	Protocol family configured on the logical interface. If the protocol is **inet**, the IP address of the interface is also displayed.	**brief**
Flags	Information about the address flag. Possible values are described in the "Addresses Flags" section under *Common Output Fields Description*.	**detail extensive** none
Destination	IP address of the remote side of the connection.	**detail extensive** none
Local	IP address of the logical interface.	**detail extensive** none
Broadcast	Broadcast address of the logical interface.	**detail extensive** none
Generation	Unique number for use by Juniper Networks technical support only.	**detail extensive**

Sample Output

show interfaces vcp (specific interface)

```
user@host> show interfaces vcp-4/0/0
Physical interface: vcp-4/0/0, Enabled, Physical link is Up
  Interface index: 128, SNMP ifIndex: 821
  Link-level type: 70, MTU: 9148, Speed: 1000mbps
  Device flags   : Present Running
  Interface flags: Promiscuous Internal: 0x4000
  Link flags     : None
  CoS queues     : 8 supported, 8 maximum usable queues
  Schedulers     : 0
  Current address: Unspecified, Hardware address: Unspecified
  Last flapped   : 2015-11-05 11:29:58 PST (10:39:25 ago)
  Input rate     : 18232 bps (26 pps)
  Output rate    : 17160 bps (25 pps)
  Active alarms  : None
  Active defects : None
  Interface transmit statistics: Disabled
```

```
  Logical interface vcp-4/0/0.32768 (Index 64) (SNMP ifIndex 822)
    Flags: Encapsulation: Unspecified
```

show interfaces vcp brief (specific interface)

```
user@host> show interfaces vcp-4/0/0 brief
Physical interface: vcp-4/0/0, Enabled, Physical link is Up
  Link-level type: 70, MTU: 9148, Speed: 1000mbps
  Device flags   : Present Running
  Interface flags: Promiscuous Internal: 0x4000
  Link flags     : None

  Logical interface vcp-4/0/0.32768
    Flags: Down Encapsulation: Unspecified
```

show interfaces vcp detail (specific interface)

```
user@host> show interfaces vcp-4/0/0 detail
Physical interface: vcp-4/0/0, Enabled, Physical link is Up
  Interface index: 128, SNMP ifIndex: 821, Generation: 131
  Link-level type: 70, MTU: 9148, Speed: 1000mbps
  Device flags   : Present Running
  Interface flags: Promiscuous Internal: 0x4000
  Link flags     : None
  CoS queues     : 8 supported, 8 maximum usable queues
  Schedulers     : 0
  Hold-times     : Up 0 ms, Down 0 ms
  Current address: Unspecified, Hardware address: Unspecified
  Last flapped   : 2015-11-05 11:29:58 PST (10:40:23 ago)
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :             74190791                12624 bps
   Output bytes  :             72984813                12080 bps
   Input  packets:              1050663                   26 pps
   Output packets:              1041055                   26 pps
   IPv6 transit statistics:
    Input  bytes  :                   0
    Output bytes  :                   0
    Input  packets:                   0
    Output packets:                   0
  Egress queues: 8 supported, 4 in use
  Queue counters:       Queued packets  Transmitted packets     Dropped packets

    0 best-effort

                                    698                  698
0
    1 expedited-forwarding

                                      0                    0
0
    2 assured-forwarding

                                      0                    0
0
    3 network-control
```

```
                                1040348              1040348
0
  Queue number:         Mapped forwarding classes
    0                   best-effort
    1                   expedited-forwarding
    2                   assured-forwarding
    3                   network-control
  Active alarms  : None
  Active defects : None
  Interface transmit statistics: Disabled

  Logical interface vcp-4/0/0.32768 (Index 64) (SNMP ifIndex 822) (Generation
129)
    Flags: Down Encapsulation: Unspecified
```

show services accounting status

Syntax show services accounting status
<inline-jflow fpc-slot *slot-number* | name (* | all | *service-name*)>

Release Information Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 13.2R2 for EX Series switches.

Description Display available Physical Interface Cards (PICs) for accounting services.

Options **none**—Display available PICs for all accounting services.

inline-jflow fpc-slot *slot-number*—(Optional) Display inline flow accounting status for the specified FPC. For a two-member MX Series Virtual Chassis or EX9200 Virtual Chassis, the master router or switch uses FPC slot numbers 0 through 11 with no offset; the backup router or switch uses FPC slot numbers 12 through 23, with an offset of 12.

name (* | all | *service-name*)—(Optional) Display available PICs. Use a wildcard character, specify all services, or provide a specific services name.

Required Privilege Level view

Related Documentation
- *show services accounting flow*
- Inline Flow Monitoring for Virtual Chassis Overview on page 191

List of Sample Output
show services accounting status name (Monitoring PIC Interface) on page 301
show services accounting status name (Service PIC Interface) on page 302
show services accounting status inline-jflow fpc-slot (When Both IPv4 and IPv6 Are Configured) on page 302
show services accounting status inline-jflow (MX80 Router When Both IPv4 and IPv6 Are Configured) on page 302
show services accounting status inline-jflow fpc-slot (PTX1000 Router When Both IPv4 and IPv6 Are Configured) on page 302

Output Fields Table 30 on page 300 lists the output fields for the **show services accounting status** command. Output fields are listed in the approximate order in which they appear.

Table 30: show services accounting status Output Fields

Field	Field Description
Service Accounting interface	Name of the service accounting interface.

Table 30: show services accounting status Output Fields *(continued)*

Field	Field Description
Service name	Name of a service that was configured at the [**edit-forwarding-options accounting**] hierarchy level. The default display,**(default sampling)**, indicates the service was configured at the [**edit-forwarding-options sampling-level**] hierarchy level.
FPC Slot	Slot number of the FPC for which the flow information is displayed.
Local interface index	Index counter of the local interface.
Interface state	Accounting state of the passive monitoring interface. • **Accounting**—PIC is actively accounting. • **Disabled**—PIC has been disabled from the CLI. • **Not accounting**—PIC is up but not accounting. This can happen while the PIC is coming online, or when the PIC is up but has no logical unit configured under the physical interface. • Unknown
Group index	Integer that represents the monitoring group of which the PIC is a member. **Group index** is a mapping from the group name to an index. It is not related to the number of monitoring groups.
Export interval (in seconds)	Configured export interval for cflowd records, in seconds.
Export format	Configured export format.
Protocol	Protocol the PIC is configured to monitor.
Engine type	Configured engine type that is inserted in output cflowd packets.
Engine ID	Configured engine ID that is inserted in output cflowd packets.
Route Record Count	Number of routes recorded.
AS Record Count	Number of autonomous systems recorded.
Route Records Set	Status of route recording; whether routes are recorded or not.
Configuration Set	Status of monitoring configuration; whether monitoring configuration is set or not.

Sample Output

show services accounting status name (Monitoring PIC Interface)

```
user@host> show services accounting status name count1
Service Accounting interface: mo-2/0/0, Local interface index: 468
Service name: count1
Interface state: Accounting
  Group index: 0
  Export interval (in seconds): 60, Export format: cflowd v8
  Protocol: IPv4, Engine type: 55, Engine ID: 5
```

Sample Output

show services accounting status name (Service PIC Interface)

```
user@host> show services accounting status name
Service Accounting interface: sp-0/1/0
Interface state: Accounting
  Export format: 9, Route record count: 0
  IFL to SNMP index count: 7, AS count: 0
  Configuration set: Yes, Route record set: No, IFL SNMP map set: Yes

Service Accounting interface: sp-1/0/0
Interface state: Accounting
  Export format: 9, Route record count: 33
  IFL to SNMP index count: 7, AS count: 1
  Configuration set: Yes, Route record set: Yes, IFL SNMP map set: Yes
```

show services accounting status inline-jflow fpc-slot (When Both IPv4 and IPv6 Are Configured)

```
user@host> show services accounting status inline-jflow fpc-slot 5
FPC Slot: 5
    IPV4 export format: Version-IPFIX, IPV6 export format: Version-IPFIX
    VPLS export format: Not set
    IPv4 Route Record Count: 5, IPv6 Route Record Count: 7
    Route Record Count: 12, AS Record Count: 1
    Route-Records Set: Yes, Config Set: Yes
```

show services accounting status inline-jflow (MX80 Router When Both IPv4 and IPv6 Are Configured)

```
user@host> show services accounting status inline-jflow

Status information
    TFEB Slot: 0
    Export format: IP-FIX
    IPv4 Route Record Count: 6, IPv6 Route Record Count: 8
    Route Record Count: 14, AS Record Count: 1
    Route-Records Set: Yes, Config Set: Yes
```

show services accounting status inline-jflow fpc-slot (PTX1000 Router When Both IPv4 and IPv6 Are Configured)

```
user@host> show services accounting status inline-jflow fpc-slot 0
Status information
FPC Slot: 0
IPV4 export format: Version-IPFIX, IPV6 export format: Version-IPFIX
VPLS export format: Not set, MPLS export format: Not set
IPv4 Route Record Count: 23, IPv6 Route Record Count: 3, MPLS Route Record Count:
 0
Route Record Count: 26, AS Record Count: 1
Route-Records Set: Yes, Config Set: Yes
IPv4 MAX FLOW Count: 0, IPv6 MAX FLOW Count: 0
VPLS MAX FLOW Count: 0, MPLS MAX FLOW Count: 2
```

show virtual-chassis active-topology (MX Series Virtual Chassis)

Syntax show virtual-chassis active-topology
<(all-members | local | member *member-id*)>

Release Information Command introduced in Junos OS Release 11.2.

Description Display information about neighbor reachability from each member router in an MX Series Virtual Chassis configuration. You can issue the **show virtual-chassis active-topology** command from the console of either member router in the Virtual Chassis.

Options

all-members—(Optional) Display neighbor reachability information for both member routers in a Virtual Chassis configuration. This is the default behavior if no options are specified.

local—(Optional) Display neighbor reachability information for the member router on which you are issuing the command.

member *member-id*—(Optional) Display neighbor reachability information for the specified Virtual Chassis member router. Replace ***member-id*** with the value 0 or 1.

Required Privilege Level view

Related Documentation

- Verifying Neighbor Reachability for Member Routers or Switches in a Virtual Chassis on page 184

List of Sample Output

show virtual-chassis active-topology all-members on page 304
show virtual-chassis active-topology local on page 304
show virtual-chassis active-topology member 1 on page 304

Output Fields Table 31 on page 303 lists the output fields for the **show virtual-chassis active-topology** command. Output fields are listed in the approximate order in which they appear.

Table 31: show virtual-chassis active-topology Output Fields

Field Name	Field Description
member*n*	Member ID of the Virtual Chassis member router for which output is displayed.
Destination ID	Member ID of the destination (neighbor) router.
Next-hop	Member ID and Virtual Chassis port interface (in the format **vcp-*slot/pic/port.logical-unit-number***) of the next-hop to which the router forwards packets for the destination ID.

Sample Output

show virtual-chassis active-topology all-members

```
{master:member0-re0}

user@host> show virtual-chassis active-topology all-members
member0:
--------------------------------------------------------------------------------
  Destination ID        Next-hop

  1                    1(vcp-5/0/0.32768)

member1:
--------------------------------------------------------------------------------
  Destination ID        Next-hop

  0                    0(vcp-1/3/0.32768)
```

show virtual-chassis active-topology local

```
{master:member0-re0}

user@host> show virtual-chassis active-topology local

  Destination ID        Next-hop

  1                    1(vcp-5/0/0.32768)
```

show virtual-chassis active-topology member 1

```
{master:member0-re0}

user@host> show virtual-chassis active-topology member 1
member1:
--------------------------------------------------------------------------------
  Destination ID        Next-hop

  0                    0(vcp-1/3/0.32768)
```

show virtual-chassis device-topology (MX Series Virtual Chassis)

Syntax show virtual-chassis device-topology
<(all-members | local | member *member-id*)>

Release Information Command introduced in Junos OS Release 11.2.

Description Display information about neighbor reachability for each hardware device in an MX Series Virtual Chassis configuration. On the MX Series router, there is only one active device for each member router. You can issue the **show virtual-chassis device-topology** command from the console of either member router in the Virtual Chassis.

Options

all-members—(Optional) Display neighbor reachability information for the device in both member routers in a Virtual Chassis configuration.

local—(Optional) Display neighbor reachability information for the device in the member router on which you are issuing the command. This is the default behavior if no options are specified.

member *member-id*—(Optional) Display neighbor reachability information for the device in the specified Virtual Chassis member router. Replace ***member-id*** with the value 0 or 1.

Required Privilege Level view

Related Documentation

- Verifying Neighbor Reachability for Hardware Devices in a Virtual Chassis on page 184

List of Sample Output

show virtual-chassis device-topology all-members on page 306
show virtual-chassis device-topology local on page 306
show virtual-chassis device-topology member 1 on page 306

Output Fields Table 32 on page 305 lists the output fields for the **show virtual-chassis device-topology** command. Output fields are listed in the approximate order in which they appear.

Table 32: show virtual-chassis device-topology Output Fields

Field Name	Field Description
member*n*	Member ID of the Virtual Chassis member router for which output is displayed.
Member	Identifier assigned to the member router in the preprovisioned Virtual Chassis configuration.
Device	Identifier assigned to the device in the member router. Because there is only one active device per member router in an MX Series Virtual Chassis configuration, the values in the Device and Member fields are identical.

Table 32: show virtual-chassis device-topology Output Fields *(continued)*

Field Name	Field Description
Status	Status of the device: • **Prsnt**—Device is currently connected to the Virtual Chassis. • **NotPrsnt**—Device is not currently connected to the Virtual Chassis.
System ID	Unique identifier derived from the device's media access control (MAC) address. The System ID is included in each Virtual Chassis Control Protocol (VCCP) packet to identify the packet owner to all members of the Virtual Chassis.
Neighbor List Member/Device/Interface	Member IDs, Device IDs, and Virtual Chassis port interfaces (in the format **vcp-*slot/pic/port***) to which this device is connected.

Sample Output

show virtual-chassis device-topology all-members

```
{master:member0-re0}

user@host> show virtual-chassis device-topology all-members
member0:
--------------------------------------------------------------------------
                                                Neighbor List
Member  Device  Status   System ID         Member  Device  Interface
   0       0    Prsnt    b0c6.9abf.6800       1       1    vcp-5/0/0
   1       1    Prsnt    001d.b510.0800       0       0    vcp-1/3/0

member1:
--------------------------------------------------------------------------
                                                Neighbor List
Member  Device  Status   System ID         Member  Device  Interface
   0       0    Prsnt    b0c6.9abf.6800       1       1    vcp-5/0/0
   1       1    Prsnt    001d.b510.0800       0       0    vcp-1/3/0
```

show virtual-chassis device-topology local

```
{master:member0-re0}

user@host> show virtual-chassis device-topology local

                                                Neighbor List
Member  Device  Status   System ID         Member  Device  Interface
   0       0    Prsnt    b0c6.9abf.6800       1       1    vcp-5/0/0
   1       1    Prsnt    001d.b510.0800       0       0    vcp-1/3/0
```

show virtual-chassis device-topology member 1

```
{master:member0-re0}

user@host> show virtual-chassis device-topology member 1
member1:
--------------------------------------------------------------------------
                                                Neighbor List
Member  Device  Status   System ID         Member  Device  Interface
```

```
0       0    Prsnt    b0c6.9abf.6800       1       1    vcp-5/0/0
1       1    Prsnt    001d.b510.0800       0       0    vcp-1/3/0
```

show virtual-chassis heartbeat (MX Series Virtual Chassis)

Syntax

show virtual-chassis heartbeat
<(brief | detail)>
<(all-members | local | member *member-id*)>

Release Information

Command introduced in Junos OS Release 14.1.

Description

Display the state of one or both member routers in an MX Series Virtual Chassis configuration when an IP-based heartbeat packet connection is configured. You can issue the **show virtual-chassis heartbeat** command from the console of either member router in the Virtual Chassis.

Options

none—Display the heartbeat state of both member routers in an MX Series Virtual Chassis.

brief | detail—(Optional) Display the specified level of output. Using the **brief** option is equivalent to issuing the command with no options (the default). The **detail** option provides more output than the **brief** option.

all-members—(Optional) Display the heartbeat state of both member routers in an MX Series Virtual Chassis. This is the default behavior if no options are specified.

local—(Optional) Display the heartbeat state of the member router from which you are issuing the command.

member *member-id*—(Optional) Display the heartbeat state of the specified member router. Replace ***member-id*** with the value 0 or 1.

Required Privilege Level

view

Related Documentation

- Verifying and Managing the Virtual Chassis Heartbeat Connection on page 190
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet on page 196
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets on page 207

List of Sample Output

show virtual-chassis heartbeat on page 310
show virtual-chassis heartbeat detail on page 310

Output Fields

Table 33 on page 309 lists the output fields for the **show virtual-chassis heartbeat** command. Output fields are listed in the approximate order in which they appear.

Table 33: show virtual-chassis heartbeat Output Fields

Field Name	Field Description	Level of Output
member*n*	Member ID of the Virtual Chassis member router for which output is displayed.	All levels
Local	IP address of the local member router that forms the heartbeat connection.	All levels
Remote	IP address of the remote member router that forms the heartbeat connection.	All levels
State	State of the heartbeat connection: • **Alive**—Master Routing Engine in the specified member router is connected and has received a heartbeat response message. • **Connected**—Master Routing Engine in the specified member router is connected and awaiting a heartbeat response message. • **Conn-Pending**—Master Routing Engine in the specified member router is connecting to the other member router. • **Detected**—Master Routing Engine in the specified member router has successfully exchanged a heartbeat connection message with the other member router when an adjacency disruption or split occurs in the Virtual Chassis. The **Detected** state persists until the heartbeat connection is reset, or until the Virtual Chassis forms again and a master router (protocol master) and backup router (protocol backup) are elected. • **Idle**—Heartbeat connection is enabled but not yet connected. • **Inactive**—Heartbeat connection is not active. To activate the connection, you must configure the heartbeat address and, optionally, the heartbeat timeout value.	All levels
Time	Date and time, in the format ***yyyy-mm-dd hh:mm:ss***, of the last connection state change. If you have cleared or never collected timestamps, this field displays **Never** and the time interval displays **0** (zero).	All levels
Heartbeat Statistics	Detailed statistics for the heartbeat connection: • **Heartbeats sent**—Number of heartbeat request messages sent. • **Heartbeats received**—Number of heartbeat response messages received. • **Heartbeats lost/missed**—Calculated difference between **Heartbeats sent** value and **Heartbeats received** value. • **Last time sent**—Date and time, in the format ***yyyy-mm-dd hh:mm:ss***, and time since, in the format **(*hh:mm:ss* ago)**, the last heartbeat request message was sent. If you have cleared or never collected timestamps, this field displays **Never** and the time interval displays **0** (zero). • **Last time received**—Date and time, in the format ***yyyy-mm-dd hh:mm:ss***, and time since, in the format **(*hh:mm:ss* ago)**, the last heartbeat request response message was received. If you have cleared or never collected timestamps, this field displays **Never** and the time interval displays **0** (zero). • **Maximum latency**—Maximum number of seconds that elapse on the local member router between transmission of a heartbeat request message and receipt of a heartbeat response message. • **Minimum latency**—Minimum number of seconds that elapse on the local member router between transmission of a heartbeat request message and receipt of a heartbeat response message.	**detail**

Sample Output

show virtual-chassis heartbeat

```
{master:member0-re0}

user@host> show virtual-chassis heartbeat
member0:
--------------------------------------------------------------------------
Local        Remote          State       Time
10.4.2.210   10.4.2.100      Alive       2014-01-14 14:46:52 PDT

member1:
--------------------------------------------------------------------------
Local        Remote          State       Time
10.4.2.100   10.4.2.210      Alive       2014-01-14 14:46:53 PDT
```

show virtual-chassis heartbeat detail

```
{master:member0-re0}

user@host> show virtual-chassis heartbeat detail
member0:
--------------------------------------------------------------------------
Local        Remote          State       Time
10.4.2.210   10.4.2.100      Alive       2014-01-14 14:46:52 PDT

Heartbeat statistics
 Heartbeats sent: 8812
 Heartbeats received: 8806
 Heartbeats lost/missed: 6
 Last time sent: 2014-01-14 14:28:00 PST (00:00:01 ago)
 Last time received: 2014-01-14 14:28:00 PST (00:00:01 ago)
 Maximum latency (secs): 1
 Minimum latency (secs): 0

member1:
--------------------------------------------------------------------------
Local        Remote          State       Time
10.4.2.100   10.4.2.210      Alive       2014-01-14 14:46:53 PDT

Heartbeat statistics
 Heartbeats sent: 8812
 Heartbeats received: 8806
 Heartbeats lost/missed: 6
 Last time sent: 2014-01-14 14:28:00 PST (00:00:01 ago)
 Maximum latency (secs): 1
 Minimum latency (secs): 0
```

show virtual-chassis protocol adjacency (MX Series Virtual Chassis)

Syntax

show virtual-chassis protocol adjacency
<(brief | detail | extensive)>
<(all-members | local | member *member-id*)>
<*system-id*>

Release Information

Command introduced in Junos OS Release 11.2.

Description

Display the entries (neighbors) in the Virtual Chassis Control Protocol (VCCP) adjacency database for an MX Series Virtual Chassis configuration. You can issue the **show virtual-chassis protocol adjacency** command from the console of either member router in the Virtual Chassis.

Options

brief | detail | extensive—(Optional) Display the specified level of output. Using the **brief** option is equivalent to issuing the command with no options (the default). The **detail** option provides more output than the **brief** option. The **extensive** option provides complete output and is most useful for customer support personnel.

all-members—(Optional) Display the VCCP adjacency database for both member routers in a Virtual Chassis. This is the default behavior if no options are specified.

local—(Optional) Display the VCCP adjacency database for the member router on which you are issuing the command.

member *member-id*—(Optional) Display the VCCP adjacency database for the specified member router. Replace ***member-id*** with the value 0 or 1.

system-id—(Optional) Display the VCCP adjacency database for the device with the specified system ID.

Required Privilege Level

view

Related Documentation

- Viewing Information in the Virtual Chassis Control Protocol Adjacency Database on page 186

List of Sample Output

show virtual-chassis protocol adjacency all-members brief on page 313
show virtual-chassis protocol adjacency member 0 detail on page 314
show virtual-chassis protocol adjacency member 0 detail 001d.b510.0800 on page 314
show virtual-chassis protocol adjacency local extensive on page 314

Output Fields

Table 34 on page 312 lists the output fields for the **show virtual-chassis protocol adjacency** command. Output fields are listed in the approximate order in which they appear.

Table 34: show virtual-chassis protocol adjacency Output Fields

Field Name	Field Description	Level of Output
member*n*	Member ID of the Virtual Chassis member router for which output is displayed.	All levels
Interface	Name of the Virtual Chassis port interface, in the format **vcp-*slot/pic/port.logical-unit-number***.	**brief**
System	System ID of the device associated with the Virtual Chassis port interface. The System ID is derived from the device's media access control (MAC) address.	**brief**
State	State of the adjacency: • **Up**—The adjacency is up. • **Down**—The adjacency is down.	All levels
Hold (secs)	Remaining hold time of the adjacency, in seconds.	**brief**
system-id	System ID of the device associated with the Virtual Chassis port interface. The System ID is derived from the device's media access control (MAC) address.	**detail**, **extensive**
interface-name	Name of the Virtual Chassis port interface, in the format **vcp-*slot/pic/port.logical-unit-number***.	**detail**, **extensive**
Expires in	Number of seconds until the adjacency expires.	**detail**, **extensive**
Priority	Priority to become the designated intermediate system.	**detail**, **extensive**
Up/Down transitions	Count of adjacency status changes from **Up** to **Down** or from **Down** to **Up**.	**detail**, **extensive**
Last transition	Time of the last **Up**/**Down** transition.	**detail**, **extensive**

Table 34: show virtual-chassis protocol adjacency Output Fields *(continued)*

Field Name	Field Description	Level of Output

Transition log	List of recent adjacency transitions, including:
	• **When**—Date and time at which a VCCP adjacency transition occurred.
	• **State**—Current state of the VCCP adjacency:
	  • **Up**—Adjacency is up and operational.
	  • **Down**—Adjacency is down and not available.
	  • **Rejected**—Adjacency has been rejected.
	• **Event**—Type of transition that occurred:
	  • **Seenself**—Possible routing loop has been detected.
	  • **Interface down**—Virtual Chassis port interface has gone down and is no longer available.
	  • **Error**—Adjacency error.
	• **Down reason**—Reason that a VCCP adjacency is down:
	  • **3-Way Handshake Failed**—Connection establishment failed.
	  • **Address Mismatch**—Address mismatch caused link failure.
	  • **Aged Out**—Link expired.
	  • **ISO Area Mismatch**—VCCP area mismatch caused link failure.
	  • **Bad Hello**—Unacceptable hello message caused link failure.
	  • **BFD Session Down**—Bidirectional failure detection caused link failure.
	  • **Interface Disabled**—Virtual Chassis port interface is disabled.
	  • **Interface Down**—Virtual Chassis port interface is unavailable.
	  • **Interface Level Disabled**—VCCP level is disabled.
	  • **Level Changed**—VCCP level has changed on the adjacency.
	  • **Level Mismatch**—Levels on adjacency are not compatible.
	  • **MPLS LSP Down**—Label-switched path (LSP) is unavailable.
	  • **MT Topology Changed**—VCCP topology has changed.
	  • **MT Topology Mismatch**—VCCP topology is mismatched.
	  • **Remote System ID Changed**—Adjacency peer system ID changed.
	  • **Protocol Shutdown**—VCCP is disabled.
	  • **CLI Command**—Adjacency brought down by user.
	  • **Unknown**—Unknown.
		extensive

Sample Output

show virtual-chassis protocol adjacency all-members brief

```
{master:member0-re0}

user@host> show virtual-chassis protocol adjacency all-members brief
member0:
--------------------------------------------------------------------------
Interface             System         State       Hold (secs)
vcp-5/0/0.32768       001d.b510.0800 Up                  57

member1:
--------------------------------------------------------------------------
```

```
Interface             System         State       Hold (secs)
vcp-1/3/0.32768       b0c6.9abf.6800 Up                  58
```

show virtual-chassis protocol adjacency member 0 detail

```
{master:member0-re0}

user@host> show virtual-chassis protocol adjacency member 0 detail
member0:
--------------------------------------------------------------------------

001d.b510.0800
  interface-name: vcp-5/0/0.32768, State: Up, Expires in 57 secs
  Priority: 0, Up/Down transitions: 1, Last transition: 18:50:41 ago
```

show virtual-chassis protocol adjacency member 0 detail 001d.b510.0800

```
{master:member0-re0}

user@host> show virtual-chassis protocol adjacency member 0 detail 001d.b510.0800
member0:
--------------------------------------------------------------------------

001d.b510.0800
  interface-name: vcp-5/0/0.32768, State: Up, Expires in 58 secs
  Priority: 0, Up/Down transitions: 1, Last transition: 18:52:08 ago
```

show virtual-chassis protocol adjacency local extensive

```
{master:member0-re0}

user@host> show virtual-chassis protocol adjacency local extensive


001d.b510.0800
  interface-name: vcp-5/0/0.32768, State: Up, Expires in 59 secs
  Priority: 0, Up/Down transitions: 1, Last transition: 18:52:40 ago
  Transition log:
  When                  State        Event           Down reason
  Mon Sep 20 17:26:44   Up           Seenself
```

show virtual-chassis protocol database (MX Series Virtual Chassis)

Syntax

show virtual-chassis protocol database
<(brief | detail | extensive)>
<(all-members | local | member *member-id*)>
<*system-id*>

Release Information

Command introduced in Junos OS Release 11.2.

Description

Display the entries in the Virtual Chassis Control Protocol (VCCP) link-state database for an MX Series Virtual Chassis configuration. The VCCP link-state database contains information about protocol data unit (PDU) packets. You can issue the **show virtual-chassis protocol database** command from the console of either member router in the Virtual Chassis.

Options

brief | detail | extensive—(Optional) Display the specified level of output. Using the **brief** option is equivalent to issuing the command with no options (the default). The **detail** option provides more output than the **brief** option. The **extensive** option provides complete output and is most useful for customer support personnel.

all-members—(Optional) Display the VCCP link-state database for both member routers in a Virtual Chassis. This is the default behavior if no options are specified.

local—(Optional) Display the VCCP link-state database for the member router on which you are issuing the command.

member *member-id*—(Optional) Display the VCCP link-state database for the specified member router. Replace ***member-id*** with the value 0 or 1.

system-id—(Optional) Display the VCCP link-state database for the neighbor with the specified system ID.

Required Privilege Level

view

Related Documentation

- Viewing Information in the Virtual Chassis Control Protocol Link-State Database on page 187

List of Sample Output

show virtual-chassis protocol database all-members brief on page 317
show virtual-chassis protocol database member 0 detail on page 318
show virtual-chassis protocol database member 0 b0c6.9abf.6800 detail on page 318
show virtual-chassis protocol database member 0 extensive on page 318

Output Fields

Table 35 on page 316 lists the output fields for the **show virtual-chassis protocol database** command. Output fields are listed in the approximate order in which they appear.

Table 35: show virtual-chassis protocol database Output Fields

Field Name	Field Description	Level of Output
member*n*	Member ID of the Virtual Chassis member router for which output is displayed.	All levels
LSP ID	Link-state PDU (LSP) identifier.	All levels
Sequence	Sequence number of the link-state PDU.	All levels
Checksum	Checksum value of the link-state PDU.	All levels
Lifetime	Remaining lifetime of the link-state PDU, in seconds.	All levels
***number* LSPs**	Total number of link-state PDUs in the specified link-state database.	none, **brief**
Neighbor, Neighbor Info	Media access control (MAC) address of the neighbor on the advertising system.	**detail**, **extensive**
Interface	Name of the Virtual Chassis port interface, in the format **vcp-*slot/pic/port.logical-unit-number***.	**detail**, **extensive**
Metric	Metric value of the prefix or neighbor.	**detail**, **extensive**
Header	Link-state PDU (LSP) packet header: • **LSP ID**—LSP identifier in the header. • **Length**—Header length, in bytes. • **Allocated length**—Length available for the header, in bytes. • **Remaining lifetime**—Remaining lifetime of the link-state PDU, in seconds. • **Interface**—The interface from which the LSP is received. • **Estimated free bytes**—Estimated number of available bytes in the LSP. • **Actual free bytes**—Actual number of available bytes in the LSP. • **Aging timer expires in**—Remaining lifetime of the LSP, in seconds.	**extensive**
Packet	Link-state PDU (LSP) packet: • **LSP ID**—Identifier for the link-state packet. • **Length**—Packet length, in bytes. • **Lifetime**—Remaining lifetime, in seconds. • **Checksum**—Checksum of the link-state PDU. • **Sequence**—Sequence number of the link-state PDU. This number increments whenever the link-state PDU is updated. • **Fixed length**—Set length for the packet, in bytes. • **Version**—Protocol version. • **Sysid length**—Length of the system ID, in bytes. The value 0 represents 6 bytes. • **Packet type**—Protocol data unit (PDU) type of the LSP. • **SW version**—Junos OS Release number.	**extensive**

Table 35: show virtual-chassis protocol database Output Fields *(continued)*

Field Name	Field Description	Level of Output
TLVs	Link-state PDU (LSP) type, length, and value (TLV): • **Member ID**—Identifier configured for the Virtual Chassis member router. • **VC ID**—Identifier assigned to the Virtual Chassis member router. • **Flags**—Internal flags that keep track of the member state for the purpose of mastership election in the Virtual Chassis. • **Priority**—Priority value associated with the role assigned to a member router in the preprovisioned Virtual Chassis configuration. For example, the priority value for the **routing-engine** role is 129. The priority value is used for mastership election in the Virtual Chassis. • **System ID**—System ID of the device associated with the Virtual Chassis port interface. The System ID is derived from the device's media access control (MAC) address. • **Device ID**—Identifier for the device; usually the same as the Member ID. • **Neighbor Info**—System ID, Virtual Chassis port interface, and metric value for VCCP neighbor. • **Topology Info**—System ID of the VCCP neighbor. • **IRI Addr Info**—Internal routing interface (IRI) IP address, which is reserved for internal communication. • **Master Info**—System ID of the master router in the Virtual Chassis. • **Backup Info**—System ID of the backup router in the Virtual Chassis. • **Stable State Info**—Internal state information used for mastership election in the Virtual Chassis. • **Member Info**—System ID, Member ID, and role of each member router in the Virtual Chassis. • **Provision Info**—Member ID and chassis serial number specified for each member router in the preprovisioned configuration for an MX Series Virtual Chassis. • **Unknown TLV**—Type and length of TLVs with unsupported content received on this device.	**extensive**
***number* queued**	Number of link-state PDUs queued on the specified Virtual Chassis port interface.	**extensive**

Sample Output

show virtual-chassis protocol database all-members brief

```
{master:member1-re0}

user@host> show virtual-chassis protocol database all-members brief
member0:
--------------------------------------------------------------------------
LSP ID                      Sequence Checksum Lifetime
001d.b510.0800.00-00           0x9eb   0xb8f1      115
b0c6.9abf.6800.00-00           0x9ee   0x8f35      116
  2 LSPs

member1:
--------------------------------------------------------------------------
LSP ID                      Sequence Checksum Lifetime
001d.b510.0800.00-00           0x9eb   0xb8f1      117
```

```
b0c6.9abf.6800.00-00           0x9ee   0x8f35      114
  2 LSPs
```

show virtual-chassis protocol database member 0 detail

```
{master:member1-re0}

user@host> show virtual-chassis protocol database member 0 detail
member0:
--------------------------------------------------------------------------

001d.b510.0800.00-00 Sequence: 0x9f5, Checksum: 0x5b2b, Lifetime: 116 secs
   Neighbor: b0c6.9abf.6800.00  Interface: vcp-1/3/0.32768 Metric:      15

b0c6.9abf.6800.00-00 Sequence: 0x9f8, Checksum: 0x326e, Lifetime: 117 secs
   Neighbor: 001d.b510.0800.00  Interface: vcp-5/0/0.32768 Metric:      15
```

show virtual-chassis protocol database member 0 b0c6.9abf.6800 detail

```
{master:member1-re0}

user@host> show virtual-chassis protocol database member 0 b0c6.9abf.6800 detail
member0:
--------------------------------------------------------------------------

b0c6.9abf.6800.00-00 Sequence: 0xa06, Checksum: 0x925b, Lifetime: 117 secs
   Neighbor: 001d.b510.0800.00  Interface: vcp-5/0/0.32768 Metric:      15
```

show virtual-chassis protocol database member 0 extensive

```
{master:member1-re0}

user@host> show virtual-chassis protocol database member 0 extensive
member0:
--------------------------------------------------------------------------

001d.b510.0800.00-00 Sequence: 0xa09, Checksum: 0xa696, Lifetime: 116 secs
   Neighbor: b0c6.9abf.6800.00  Interface: vcp-1/3/0.32768 Metric:      15

  Header: LSP ID: 001d.b510.0800.00-00, Length: 804 bytes
    Allocated length: 804 bytes,
    Remaining lifetime: 116 secs, Interface: 64
    Estimated free bytes: 0, Actual free bytes: 0
    Aging timer expires in: 116 secs

  Packet: LSP ID: 001d.b510.0800.00-00, Length: 804 bytes, Lifetime : 118 secs
    Checksum: 0xa696, Sequence: 0xa09,
    Fixed length: 27 bytes, Version: 1, Sysid length: 0 bytes
    Packet type: 18, SW version: 11.1

  TLVs:
    Node Info: Member ID: 1, VC ID: 5a6a.e747.8511, Flags: 3, Priority: 129
       System ID: 001d.b510.0800, Device ID: 1
    Unknown TLV, Type: 0, Length: 0
    ...
    Unknown TLV, Type: 0, Length: 0
    Unknown TLV, Type: 1, Length: 1
    Neighbor Info: b0c6.9abf.6800.00, Interface: vcp-1/3/0.32768, Metric: 15
    Topology Info: System ID: 001d.b510.0800,
    Topology Info: System ID: b0c6.9abf.6800,
    IRI Addr Info: IP Address: 128.0.0.1,
```

```
    IRI Addr Info: IP Address: 128.0.0.4,
    IRI Addr Info: IP Address: 128.0.0.5,
    IRI Addr Info: IP Address: 128.0.0.6,
    IRI Addr Info: IP Address: 128.0.0.17,
    Master Info: System ID: 001d.b510.0800
    Backup Info: System ID: b0c6.9abf.6800
    Stable State Info: Master ID: 001d.b510.0800, Backup ID: b0c6.9abf.6800
    Member Info: System ID: b0c6.9abf.6800, Member ID: 0 Member role: Backup
       System ID: b0c6.9abf.6800, Device ID: 0
    Member Info: System ID: 001d.b510.0800, Member ID: 1 Member role: Master
       System ID: 001d.b510.0800, Device ID: 1
    Provision Info: Member ID: 1 Serial Number: JN10C78D1AFC,
    Provision Info: Member ID: 0 Serial Number: JN115FDADAFB,
    Unknown TLV, Type: 24, Length: 1
    Unknown TLV, Type: 28, Length: 56

  1 queued :
  Send PSN on vcp-5/0/0.32768 for 00:00:01

b0c6.9abf.6800.00-00 Sequence: 0xa0d, Checksum: 0x82d2, Lifetime: 118 secs
   Neighbor: 001d.b510.0800.00  Interface: vcp-5/0/0.32768 Metric:      15

  Header: LSP ID: b0c6.9abf.6800.00-00, Length: 808 bytes
    Allocated length: 1400 bytes,
    Remaining lifetime: 118 secs, Interface: 0
    Estimated free bytes: 546, Actual free bytes: 592
    Aging timer expires in: 118 secs

  Packet: LSP ID: b0c6.9abf.6800.00-00, Length: 808 bytes, Lifetime : 118 secs
    Checksum: 0x82d2, Sequence: 0xa0d,
    Fixed length: 27 bytes, Version: 1, Sysid length: 0 bytes
    Packet type: 18, SW version: 11.1

  TLVs:
    Node Info: Member ID: 0, VC ID: 5a6a.e747.8511, Flags: 5, Priority: 129
       System ID: b0c6.9abf.6800, Device ID: 0
    Unknown TLV, Type: 0, Length: 0
    ...
    Unknown TLV, Type: 0, Length: 0
    Unknown TLV, Type: 1, Length: 1
    Neighbor Info: 001d.b510.0800.00, Interface: vcp-5/0/0.32768, Metric: 15
    Topology Info: System ID: 001d.b510.0800,
    Topology Info: System ID: b0c6.9abf.6800,
    IRI Addr Info: IP Address: 128.0.0.1,
    IRI Addr Info: IP Address: 128.0.0.4,
    IRI Addr Info: IP Address: 128.0.0.5,
    IRI Addr Info: IP Address: 128.0.0.6,
    IRI Addr Info: IP Address: 128.0.0.17,
    IRI Addr Info: IP Address: 128.0.0.21,
    Master Info: System ID: 001d.b510.0800
    Backup Info: System ID: b0c6.9abf.6800
    Stable State Info: Master ID: 001d.b510.0800, Backup ID: b0c6.9abf.6800
    Member Info: System ID: b0c6.9abf.6800, Member ID: 0 Member role: Backup
       System ID: b0c6.9abf.6800, Device ID: 0
    Member Info: System ID: 001d.b510.0800, Member ID: 1 Member role: Master
       System ID: 001d.b510.0800, Device ID: 1
    Provision Info: Member ID: 1 Serial Number: JN10C78D1AFC,
    Provision Info: Member ID: 0 Serial Number: JN115FDADAFB,
    Unknown TLV, Type: 24, Length: 1
    Unknown TLV, Type: 28, Length: 56
```

```
1 queued :
Retransmit on vcp-5/0/0.32768 for 00:00:01
```

show virtual-chassis protocol interface (MX Series Virtual Chassis)

Syntax

show virtual-chassis protocol interface
<(brief | detail)>
<*interface-name*>
<(all-members | local | member *member-id*)>

Release Information

Command introduced in Junos OS Release 11.2.

Description

Display Virtual Chassis Control Protocol (VCCP) information about Virtual Chassis port interfaces in an MX Series Virtual Chassis. You can issue the **show virtual-chassis protocol interface** command from the console of either member router in the Virtual Chassis.

Options

brief | detail—(Optional) Display the specified level of output. Using the **brief** option is equivalent to issuing the command with no options (the default). The **detail** option provides more output than the **brief** option.

all-members—(Optional) Display VCCP information about Virtual Chassis port interfaces for both member routers in a Virtual Chassis. This is the default behavior if no options are specified.

interface-name—(Optional) Display VCCP information about Virtual Chassis port interfaces for the specified Virtual Chassis port, in the format **vcp-*slot/pic/port.logical-unit-number***.

local—(Optional) Display VCCP information about Virtual Chassis port interfaces for the member router on which you are issuing the command.

member *member-id*—(Optional) Display VCCP information about Virtual Chassis port interfaces for the specified member router. Replace ***member-id*** with the value 0 or 1.

Required Privilege Level

view

Related Documentation

- Viewing Information About Virtual Chassis Port Interfaces in the Virtual Chassis Control Protocol Database on page 188

List of Sample Output

show virtual-chassis protocol interface brief all-members on page 322
show virtual-chassis protocol interface detail all-members on page 323
show virtual-chassis protocol interface detail local on page 323

Output Fields

Table 36 on page 322 lists the output fields for the **show virtual-chassis protocol interface** command. Output fields are listed in the approximate order in which they appear.

Table 36: show virtual-chassis protocol interface Output Fields

Field Name	Field Description	Level of Output
member*n*	Member ID of the Virtual Chassis member router for which output is displayed.	All levels
Interface	Name of the Virtual Chassis port interface, in the format **vcp-*slot/pic/port.logical-unit-number***.	**brief**
State	State of the Virtual Chassis port interface: • **Up**—The interface is up. • **Down**—The interface is down.	**brief**
Metric	Metric value for this Virtual Chassis port interface.	All levels
vcp-*slot/pic/port.logical-unit-number*	Name of the Virtual Chassis port interface.	**detail**
Index	Interface index number assigned by the Junos OS software.	**detail**
State	Internal implementation information.	**detail**
LSP interval	Interval, in milliseconds, between link-state protocol data units (PDUs) sent from the interface.	**detail**
***type* Hello padding**	Type of hello padding: • **Adaptive**—On point-to-point connections, the hello packets are padded from the initial detection of a new neighbor until the neighbor verifies the adjacency as Up in the adjacency state type, length, and value (TLV). If the neighbor does not support the adjacency state TLV, then padding continues. On LAN connections, padding starts from the initial detection of a new neighbor until there is at least one active adjacency on the interface. • **Loose**—(Default) The hello packet is padded from the initial detection of a new neighbor until the adjacency transitions to the Up state. • **Strict**—Padding is performed on all interface types and for all adjacency states, and is continuous.	**detail**
Adjacencies	Number of adjacencies established on this Virtual Chassis port interface.	**detail**
Hello(s)	Hello interval for the Virtual Chassis port interface.	**detail**
Hold(s)	Hold time for the Virtual Chassis port interface.	**detail**

Sample Output

show virtual-chassis protocol interface brief all-members

```
{master:member1-re0}

user@host> show virtual-chassis protocol interface brief all-members
```

```
member0:
--------------------------------------------------------------------------
IS-IS interface database:
Interface             State         Metric
vcp-5/0/0.32768       Up             15

member1:
--------------------------------------------------------------------------
IS-IS interface database:
Interface             State         Metric
vcp-1/3/0.32768       Up             15
```

show virtual-chassis protocol interface detail all-members

```
{master:member1-re0}

user@host> show virtual-chassis protocol interface detail all-members
member0:
--------------------------------------------------------------------------

IS-IS interface database:
vcp-5/0/0.32768
  Index: 64, State: 0x46
  LSP interval: 100 ms, Loose Hello padding
  Adjacencies Metric Hello (s) Hold (s)n           1     15         3     60

member1:
--------------------------------------------------------------------------

IS-IS interface database:
vcp-1/3/0.32768
  Index: 64, State: 0x86
  LSP interval: 100 ms, Loose Hello padding
  Adjacencies Metric Hello (s) Hold (s)n           1     15         3     60
```

show virtual-chassis protocol interface detail local

```
{master:member1-re0}

user@host> show virtual-chassis protocol interface detail local


IS-IS interface database:
vcp-1/3/0.32768
  Index: 64, State: 0x46
  LSP interval: 100 ms, Loose Hello padding
  Adjacencies Metric Hello (s) Hold (s)n           1     15         3     60
```

show virtual-chassis protocol route (MX Series Virtual Chassis)

Syntax

show virtual-chassis protocol route
<*destination-id*>
<(all-members | local | member *member-id*)>

Release Information

Command introduced in Junos OS Release 11.2.

Description

Display the Virtual Chassis Control Protocol (VCCP) unicast and multicast routing tables for an MX Series Virtual Chassis. You can issue the **show virtual-chassis protocol route** command from the console of either member router in the Virtual Chassis.

Options

all-members—(Optional) Display the VCCP unicast and multicast routing tables for both member routers in a Virtual Chassis configuration. This is the default behavior if no options are specified.

destination-id—(Optional) Display the VCCP unicast and multicast routing tables to the destination with the specified system ID.

local—(Optional) Display the VCCP unicast and multicast routing tables for the member router on which you are issuing the command.

member *member-id*—(Optional) Display the VCCP unicast and multicast routing tables for the specified member router. Replace ***member-id*** with the value 0 or 1.

Required Privilege Level

view

Related Documentation

- Viewing Virtual Chassis Control Protocol Routing Tables on page 188

List of Sample Output

show virtual-chassis protocol route all-members on page 325
show virtual-chassis protocol route member 0 001d.b510.0800 (For Specific Member ID and Destination ID) on page 326

Output Fields

Table 37 on page 324 lists the output fields for the **show virtual-chassis protocol route** command. Output fields are listed in the approximate order in which they appear.

Table 37: show virtual-chassis protocol route Output Fields

Field Name	Field Description
member*n*	Member ID of the Virtual Chassis member router for which output is displayed.
Dev	System ID of the device (member router) that stores the VCCP routing tables. The System ID is derived from the router's media access control (MAC) address.
ucast routing table	VCCP unicast routing table.

Table 37: show virtual-chassis protocol route Output Fields *(continued)*

Field Name	Field Description
mcast routing table	VCCP multicast routing table.
Current version	Version of the shortest-path-first (SPF) algorithm that generated the VCCP unicast or multicast routing table.
System ID	System ID of the device, derived from the device's MAC address.
Version	Version of the SPF algorithm that generated this route in the VCCP unicast or multicast routing table.
Metric	Metric value required to reach this device.
Interface	Name of the Virtual Chassis port interface (in the format **vcp-*slot/pic/port.logical-unit-number***) that interconnects the devices.
Via	MAC address of the next-hop device, if applicable.

Sample Output

show virtual-chassis protocol route all-members

```
{master:member1-re0}

user@host> show virtual-chassis protocol route all-members
member0:
--------------------------------------------------------------------------

Dev b0c6.9abf.6800 ucast routing table             Current version: 17
----------------
System ID          Version   Metric Interface     Via
001d.b510.0800          17       15 vcp-5/0/0.32768 001d.b510.0800
b0c6.9abf.6800          17        0

Dev b0c6.9abf.6800 mcast routing table             Current version: 17
----------------
System ID          Version   Metric Interface     Via
001d.b510.0800          17
b0c6.9abf.6800          17          vcp-5/0/0.32768

member1:
--------------------------------------------------------------------------

Dev 001d.b510.0800 ucast routing table             Current version: 17
----------------
System ID          Version   Metric Interface     Via
001d.b510.0800          17        0
b0c6.9abf.6800          17       15 vcp-1/3/0.32768 b0c6.9abf.6800

Dev 001d.b510.0800 mcast routing table             Current version: 17
----------------
System ID          Version   Metric Interface     Via
001d.b510.0800          17          vcp-1/3/0.32768
b0c6.9abf.6800          17
```

show virtual-chassis protocol route member 0 001d.b510.0800 (For Specific Member ID and Destination ID)

```
{master:member1-re0}

user@host> show virtual-chassis protocol route member 0 001d.b510.0800
member0:
--------------------------------------------------------------------------

Dev b0c6.9abf.6800 ucast routing table             Current version: 17
----------------
System ID          Version   Metric Interface     Via
001d.b510.0800          17       15 vcp-5/0/0.32768 001d.b510.0800
b0c6.9abf.6800          17        0

Dev b0c6.9abf.6800 mcast routing table             Current version: 17
----------------
System ID          Version   Metric Interface     Via
001d.b510.0800          17
b0c6.9abf.6800          17          vcp-5/0/0.32768
```

show virtual-chassis protocol statistics (MX Series Virtual Chassis)

Syntax show virtual-chassis protocol statistics
<*interface-name*>
<(all-members | local | member *member-id*)>

Release Information Command introduced in Junos OS Release 11.2.

Description Display Virtual Chassis Control Protocol (VCCP) statistics for one or both member routers, or for a specified Virtual Chassis port interface, in an MX Series Virtual Chassis. You can issue the **show virtual-chassis protocol statistics** command from the console of either member router in the Virtual Chassis.

Options

all-members—(Optional) Display VCCP statistics for both member routers in a Virtual Chassis configuration. This is the default behavior if no options are specified.

interface-name—(Optional) Display VCCP statistics for the specified Virtual Chassis port interface, in the format **vcp-*slot/pic/port.logical-unit-number***.

local—(Optional) Display VCCP statistics for the member router on which you are issuing the command.

member *member-id*—(Optional) Display VCCP statistics for the specified member router. Replace ***member-id*** with the value 0 or 1.

Required Privilege Level view

Related Documentation

- Viewing Virtual Chassis Control Protocol Statistics for Member Devices and Virtual Chassis Ports on page 189

List of Sample Output

show virtual-chassis protocol statistics all-members on page 328
show virtual-chassis protocol statistics vcp-1/3/0.32768 member 1 (For Specific Virtual Chassis Port Interface and Member ID) on page 329

Output Fields Table 38 on page 327 lists the output fields for the **show virtual-chassis protocol statistics** command. Output fields are listed in the approximate order in which they appear.

Table 38: show virtual-chassis protocol statistics Output Fields

Field Name	Field Description
member*n*	Member ID of the Virtual Chassis member router for which output is displayed.
PDU type	Type of protocol data unit (PDU).
Received	Number of PDUs received since VCCP started or since the statistics were set to zero.

Table 38: show virtual-chassis protocol statistics Output Fields *(continued)*

Field Name	Field Description
Processed	Number of PDUs received minus the number of PDUs dropped.
Drops	Number of PDUs dropped.
Sent	Number of PDUs transmitted since VCCP started or since the statistics were set to zero.
Rexmit	Number of PDUs retransmitted since VCCP started or since the statistics were set to zero.
Total packets received	Total number of PDUs received since VCCP started or since the statistics were set to zero.
Sent	Total number of PDUs transmitted since VCCP started or since the statistics were set to zero.
LSP queue length	Number of link-state PDUs waiting in the queue to be processed.
Drops	Number of link-state PDUs dropped.
SPF runs	Number of shortest-path-first (SPF) calculations performed.
Fragments rebuilt	Number of link-state PDU fragments computed by the local system.
LSP regenerations	Number of regenerated link-state PDUs. A link-state PDU is regenerated when the PDU nears the end of its lifetime and has not changed.
Purges initiated	Number of purges initiated by the software. A purge is initiated when the software determines that it must remove a link-state PDU from the network.

Sample Output

show virtual-chassis protocol statistics all-members

```
{master:member1-re0}

user@host> show virtual-chassis protocol statistics all-members
member0:
--------------------------------------------------------------------------

IS-IS statistics for b0c6.9abf.6800:
PDU type       Received    Processed        Drops         Sent       Rexmit
LSP                2937         2937            0         2934            0
HELLO              2913         2913            0         2922            0
CSNP                  1            1            0            1            0
PSNP               2916         2916            0         2925            0
Unknown               0            0            0            0            0
Totals             8767         8767            0         8782            0

Total packets received: 8767 Sent: 8782

LSP queue length: 0 Drops: 0
SPF runs: 17
```

```
Fragments rebuilt: 2955
LSP regenerations: 14
Purges initiated: 0

member1:
--------------------------------------------------------------------------

IS-IS statistics for 001d.b510.0800:
PDU type       Received    Processed        Drops         Sent       Rexmit
LSP                2934         2934            0         2937            0
HELLO              2922         2922            0         2914            0
CSNP                  1            1            0            1            0
PSNP               2925         2925            0         2916            0
Unknown               0            0            0            0            0
Totals             8782         8782            0         8768            0

Total packets received: 8782 Sent: 8768

LSP queue length: 0 Drops: 0
SPF runs: 17
Fragments rebuilt: 2953
LSP regenerations: 11
Purges initiated: 0
```

show virtual-chassis protocol statistics vcp-1/3/0.32768 member 1 (For Specific Virtual Chassis Port Interface and Member ID)

```
{master:member1-re0}

user@host> show virtual-chassis protocol statistics vcp-1/3/0.32768 member 1
member1:
--------------------------------------------------------------------------
vcp-1/3/0.32768

IS-IS statistics for 001d.b510.0800:
PDU type       Received    Processed        Drops         Sent       Rexmit
LSP                3013         3013            0         3016            0
HELLO              3001         3001            0         2993            0
CSNP                  1            1            0            1            0
PSNP               3003         3003            0         2994            0
Unknown               0            0            0            0            0
Totals             9018         9018            0         9004            0

Total packets received: 9018 Sent: 9004
```

show virtual-chassis status (MX Series Virtual Chassis)

Syntax show virtual-chassis status

Release Information Command introduced in Junos OS Release 11.2.

Description Display information about the status of both member routers in an MX Series Virtual Chassis configuration. You can issue the **show virtual-chassis status** command from the console of either member router in the Virtual Chassis.

Required Privilege Level view

Related Documentation

- Verifying the Status of Virtual Chassis Member Routers or Switches on page 183
- Verifying and Managing the Virtual Chassis Heartbeat Connection on page 190
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in the Same Subnet on page 196
- Example: Determining Member Health Using an MX Series Virtual Chassis Heartbeat Connection with Member Routers in Different Subnets on page 207

List of Sample Output

show virtual-chassis status on page 331
show virtual-chassis status (Mismatch Status) on page 331

Output Fields Table 39 on page 330 lists the output fields for the **show virtual-chassis status** command. Output fields are listed in the approximate order in which they appear.

Table 39: show virtual-chassis status Output Fields

Field Name	Field Description
Virtual Chassis ID	Assigned ID that applies to the entire Virtual Chassis configuration.
Member ID	Member ID assigned in the preprovisioned Virtual Chassis configuration, and the Flexible PIC Concentrator (FPC) slot range, including offset, for each member router in the Virtual Chassis.
Status	State of the member router: • **Heartbt**—Router has used heartbeat packet connection to maintain mastership roles during an adjacency disruption or split in the Virtual Chassis configuration. • **Mismatch**—Virtual Chassis not formed due to a mismatch in the slot count value of the member routers. In a Virtual Chassis with an MX2020 member router and either an MX960 or MX2010 member router, a mismatch status occurs if you do not set the slot count value of the MX960 router or MX2010 router to 20 to match the slot count of the MX2020 router. • **Prsnt**—Router is currently connected to the Virtual Chassis. • **NotPrsnt**—Router is not currently connected to the Virtual Chassis.

Table 39: show virtual-chassis status Output Fields *(continued)*

Field Name	Field Description
Serial No	Serial number of the member router.
Model	Model number of the member router.
Mastership priority	Metric used by the Virtual Chassis software for the mastership election algorithm. This value is assigned by the software and is not configurable in the current release.
Role	Role of the member router in the Virtual Chassis: **Master** or **Backup**. The asterisk (*) following the Role denotes the router on which the **show virtual-chassis status** command was issued.
Neighbor List ID Interface	Member IDs and Virtual Chassis port interfaces (in the format **vcp-*slot/pic/port***) to which this member router is connected.

Sample Output

show virtual-chassis status

```
{master:member1-re0}

user@host> show virtual-chassis status
Preprovisioned Virtual Chassis
Virtual Chassis ID: 5a6a.e747.8511
                                                  Mastership          Neighbor List
Member ID      Status   Serial No    Model    priority    Role      ID  Interface
0 (FPC   0- 11) Prsnt    JN115FDADAFB mx480          129  Backup     1  vcp-5/0/0
1 (FPC  12- 23) Prsnt    JN10C78D1AFC mx240          129  Master*    0  vcp-1/3/0
```

show virtual-chassis status (Mismatch Status)

```
{master:member1-re0}

user@host> show virtual-chassis status
Preprovisioned Virtual Chassis
Virtual Chassis ID: 4b43.17d9.1dcf
                                                  Mastership            Neighbor
List
Member ID      Status   Serial No    Model    priority    Role      ID  Interface
0 (FPC   0- 19) Mismatch JN11CA3F9AFK mx2010
1 (FPC  20- 39) Prsnt    JN122D1DAAFJ mx2020          129  Master*
```

show virtual-chassis vc-port (MX Series Virtual Chassis)

Syntax
show virtual-chassis vc-port
<(all-members | local | member *member-id*)>

Release Information
Command introduced in Junos OS Release 11.2.

Description
Display the operational status of the Virtual Chassis ports for both member routers or for a specified member router in an MX Series Virtual Chassis configuration. You can issue the **show virtual-chassis vc-port** command from the console of either member router in the Virtual Chassis.

Options
all-members—(Optional) Display the operational status of the Virtual Chassis ports for both member routers in a Virtual Chassis configuration.

local—(Optional) Display the operational status of the Virtual Chassis ports on the member router on which you are issuing the command. This is the default behavior if no options are specified.

member *member-id*—(Optional) Display the operational status of the Virtual Chassis ports on the specified member router. Replace ***member-id*** with the value 0 or 1.

Required Privilege Level
view

Related Documentation
- Verifying the Operation of Virtual Chassis Ports on page 183
- Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 58
- Example: Configuring Interchassis Redundancy for MX Series 3D Universal Edge Routers Using a Virtual Chassis on page 75

List of Sample Output
show virtual-chassis vc-port all-members on page 333
show virtual-chassis vc-port local on page 333
show virtual-chassis vc-port member 0 on page 334
show virtual-chassis vc-port (Packet Forwarding Engine is not Ready) on page 334

Output Fields
Table 40 on page 332 lists the output fields for the **show virtual-chassis vc-port** command. Output fields are listed in the approximate order in which they appear.

Table 40: show virtual-chassis vc-port Output Fields

Field Name	Field Description
member*n*	Member ID of the Virtual Chassis member router for which output is displayed.
Interface or Slot/PIC/Port	Location, in the format ***slot/pic/port***, of the Virtual Chassis ports configured on the member router.

Table 40: show virtual-chassis vc-port Output Fields *(continued)*

Field Name	Field Description
Type	Type of Virtual Chassis port. **Configured** indicates that the Virtual Chassis port is properly configured.
Trunk ID	Trunk ID value assigned to a link aggregation group (LAG) formed by the Virtual Chassis. A positive number indicates that a trunk exists. The value **-1** indicates that a trunk is not present.
Status	State of the Virtual Chassis port interface: • **Up**—The port interface is up. • **Up (Resync-not-set)**—The port interface is up but the packet forwarding engine is not ready for forwarding. • **Down**—The port interface is down. • **Absent**—The port interface is absent.
Speed (mbps)	Speed, in megabits per second, of the Virtual Chassis port interface.
Neighbor ID Interface	Member IDs and Virtual Chassis port interfaces (in **vcp-*slot/pic/port*** format) that are connected to this member router.

Sample Output

show virtual-chassis vc-port all-members

```
{master:member1-re0}

user@host> show virtual-chassis vc-port all-members
member0:
--------------------------------------------------------------------------
Interface       Type              Trunk  Status       Speed        Neighbor
or                                 ID                (mbps)       ID  Interface
Slot/PIC/Port
5/0/0           Configured         -1    Up           10000        1   vcp-1/3/0


member1:
--------------------------------------------------------------------------
Interface       Type              Trunk  Status       Speed        Neighbor
or                                 ID                (mbps)       ID  Interface
Slot/PIC/Port
1/3/0           Configured         -1    Up           10000        0   vcp-5/0/0
```

show virtual-chassis vc-port local

```
{master:member1-re0}

user@host> show virtual-chassis vc-port local

Interface       Type              Trunk  Status       Speed        Neighbor
or                                 ID                (mbps)       ID  Interface
Slot/PIC/Port
1/3/0           Configured         -1    Up           10000        0   vcp-5/0/0
```

show virtual-chassis vc-port member 0

```
{master:member1-re0}

user@host> show virtual-chassis vc-port member 0
member0:
--------------------------------------------------------------------------
Interface       Type              Trunk  Status       Speed        Neighbor
or                                 ID                (mbps)       ID  Interface
Slot/PIC/Port
5/0/0           Configured         -1    Up           10000        1   vcp-1/3/0
```

show virtual-chassis vc-port (Packet Forwarding Engine is not Ready)

```
{master:member1-re0}

user@host> show virtual-chassis vc-port
member1:
--------------------------------------------------------------------------
Interface      Type       Trunk  Status             Speed     Neighbor
or                        ID                        (mbps)    ID  Interface
Slot/PIC/Port
3/0/2          Configured  -1    Up              10000      1   vcp-1/1/4
3/0/3          Configured  -1    Up (Resync-not-set)   10000     1   vcp-1/1/0
```