

# Junos<sup>®</sup> OS

---

## RIFT Feature Guide

Published  
2019-12-22

Juniper Networks, Inc.  
1133 Innovation Way  
Sunnyvale, California 94089  
USA  
408-745-2000  
[www.juniper.net](http://www.juniper.net)

Juniper Networks, the Juniper Networks logo, Juniper, and Junos are registered trademarks of Juniper Networks, Inc. in the United States and other countries. All other trademarks, service marks, registered marks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

*Junos<sup>®</sup> OS RIFT Feature Guide*

Copyright © 2019 Juniper Networks, Inc. All rights reserved.

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 <https://support.juniper.net/support/eula/>. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

# Table of Contents

## About the Documentation | v

Documentation and Release Notes | v

Using the Examples in This Manual | v

    Merging a Full Example | vi

    Merging a Snippet | vii

Documentation Conventions | vii

Documentation Feedback | x

Requesting Technical Support | x

    Self-Help Online Tools and Resources | xi

    Creating a Service Request with JTAC | xi

## 1

## Overview

### RIFT Overview | 15

    Understanding Junos Implementation of Routing in Fat Tree (RIFT) Protocol | 15

        Benefits of RIFT Protocol | 15

        RIFT Protocol Overview | 16

        Impact of Junos Implementation of RIFT Protocol on Network Performance | 17

        Unsupported Features with RIFT Protocol | 17

    Enabling the RIFT Protocol | 17

## 2

## Configuration Statements

    capabilities (Protocols RIFT) | 29

    default-prefixes (Protocols RIFT) | 30

    export (Protocols RIFT) | 32

    interface (Protocols RIFT) | 34

    level (Protocols RIFT) | 37

    lie-receive-address (Protocols RIFT) | 38

    lie-transmit-address (Protocols RIFT) | 39

    rift | 40

## 3

**Operational Commands**

`show rift database` | 47

`show rift flood-reduction` | 51

`show rift interface` | 53

`show rift node` | 59

`show rift path-computation` | 62

`show rift routes` | 64

`show rift tie` | 68

`show rift topology` | 70

`show rift versions` | 73

# About the Documentation

## IN THIS SECTION

- Documentation and Release Notes | v
- Using the Examples in This Manual | v
- Documentation Conventions | vii
- Documentation Feedback | x
- Requesting Technical Support | x

Use this guide to integrate the Routing in Fat Tree (RIFT) routing protocol into Junos OS. RIFT is an interior gateway protocol (IGP) that is used to route packets in variants of CLOS-based and fat tree network topologies (also called the spine and leaf model).

## Documentation and Release Notes

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

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 <https://www.juniper.net/books>.

## 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.xml; }
```

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 viii](#) 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 viii defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
<b>Bold text like this</b>	Represents text that you type.	To enter configuration mode, type the <b>configure</b> command:  user@host> <b>configure</b>
Fixed-width text like this	Represents output that appears on the terminal screen.	user@host> <b>show chassis alarms</b>  No alarms currently active
<i>Italic text like this</i>	<ul style="list-style-type: none"> <li>Introduces or emphasizes important new terms.</li> <li>Identifies guide names.</li> <li>Identifies RFC and Internet draft titles.</li> </ul>	<ul style="list-style-type: none"> <li>A policy <i>term</i> is a named structure that defines match conditions and actions.</li> <li><i>Junos OS CLI User Guide</i></li> <li>RFC 1997, <i>BGP Communities Attribute</i></li> </ul>



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

Convention	Description	Examples
<i>Italic text like this</i>	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name:  [edit] root@# <b>set system domain-name</b> <i>domain-name</i>
<b>Text like this</b>	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	<ul style="list-style-type: none"> <li>To configure a stub area, include the <b>stub</b> statement at the [edit <b>protocols ospf area area-id</b>] hierarchy level.</li> <li>The console port is labeled <b>CONSOLE</b>.</li> </ul>
< > (angle brackets)	Encloses optional keywords or variables.	<b>stub</b> <default-metric <i>metric</i> >;
(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.	<b>broadcast   multicast</b>  ( <i>string1</i>   <i>string2</i>   <i>string3</i> )
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	<b>rsvp { # Required for dynamic MPLS only</b>
[ ] (square brackets)	Encloses a variable for which you can substitute one or more values.	<b>community name members [ <i>community-ids</i> ]</b>
Indentation and braces ( { } )	Identifies a level in the configuration hierarchy.	[edit] routing-options { static { route default { nexthop <i>address</i> ; retain; } } }
; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	

## GUI Conventions

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

Convention	Description	Examples
<b>Bold text like this</b>	Represents graphical user interface (GUI) items you click or select.	<ul style="list-style-type: none"> <li>In the Logical Interfaces box, select <b>All Interfaces</b>.</li> <li>To cancel the configuration, click <b>Cancel</b>.</li> </ul>
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select <b>Protocols&gt;Ospf</b> .

## Documentation Feedback

We encourage you to provide feedback so that we can improve our documentation. You can use either of the following methods:

- Online feedback system—Click TechLibrary Feedback, on the lower right of any page on the [Juniper Networks TechLibrary](#) site, and do one of the following:



- Click the thumbs-up icon if the information on the page was helpful to you.
- Click the thumbs-down icon if the information on the page was not helpful to you or if you have suggestions for improvement, and use the pop-up form to provide feedback.
- E-mail—Send your comments to [techpubs-comments@juniper.net](mailto: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 Juniper Care or Partner Support Services 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 <https://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf>.
- Product warranties—For product warranty information, visit <https://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: <https://www.juniper.net/customers/support/>
- Search for known bugs: <https://prsearch.juniper.net/>
- Find product documentation: <https://www.juniper.net/documentation/>
- Find solutions and answer questions using our Knowledge Base: <https://kb.juniper.net/>
- Download the latest versions of software and review release notes: <https://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications: <https://kb.juniper.net/InfoCenter/>
- Join and participate in the Juniper Networks Community Forum: <https://www.juniper.net/company/communities/>
- Create a service request online: <https://myjuniper.juniper.net>

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

## Creating a Service Request with JTAC

You can create a service request with JTAC on the Web or by telephone.

- Visit <https://myjuniper.juniper.net>.
- 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 <https://support.juniper.net/support/requesting-support/>.

# 1

CHAPTER

## Overview

---

RIFT Overview | 15

---



# RIFT Overview

## SUMMARY

Routing in Fat Tree (RIFT) is a zero OpEx routing protocol that you can use to route packets in variants of CLOS-based and fat tree network topologies. It is a hybrid of both link-state and distance-vector techniques, and provides several benefits for IP fabrics, such as ease of management, and adding resiliency to the network.

## IN THIS SECTION

- [Understanding Junos Implementation of Routing in Fat Tree \(RIFT\) Protocol | 15](#)
- [Enabling the RIFT Protocol | 17](#)

## Understanding Junos Implementation of Routing in Fat Tree (RIFT) Protocol

### IN THIS SECTION

- [Benefits of RIFT Protocol | 15](#)
- [RIFT Protocol Overview | 16](#)
- [Impact of Junos Implementation of RIFT Protocol on Network Performance | 17](#)
- [Unsupported Features with RIFT Protocol | 17](#)

### Benefits of RIFT Protocol

The RIFT protocols is a zero OpEx routing protocol that enables:

- Requires almost zero necessary configuration making IP fabrics simpler to manage.
- Extensive tracing and logging capabilities allowing scaling advantage for IP fabrics.
- Maximum utilization of paths without looping thereby adding resiliency in the IP fabric.

As a hybrid of the distance vector and link state protocols, the RIFT protocol inherits the advantages of both the protocol types, providing additional benefits such as:

- Fastest possible convergence.
- Auto-detection of topology.
- Minimal routes on top-of-rack devices.
- High degree of equal-cost multipath (ECMP).

## RIFT Protocol Overview

With the increase in deployment of IP forwarding-based data centers in CLOS and fat-tree architectures (also called the spine and leaf model), interior gateway protocols (IGPs) and BGP are currently used to handle the necessary routing decisions. The approach used by these protocols relies on complex and highly expensive operational extensions that fail to meet the requirements of such IP fabrics. This is because the IGP and BGP protocols were originally built for generic and sparse network topologies. Routing in Fat Trees (RIFT) overcomes these issues and meets the needs of evolving IP fabrics.

The RIFT protocol is an open standard protocol. It is a hybrid version of a distance vector protocol that uses diffused computation toward the leafs, and a link state protocol that uses distributed computation and flooding toward the spines. In other words, with the RIFT protocol enabled, devices flood their link-state information in the northern direction, while every switch except the leafs generate a default route (under normal conditions), which is flooded in the southern direction.

The key features of the RIFT protocol:

- Automatic construction of fat-tree topologies.
- Automatic detection of miscabled links of the IP fabric.
- Minimizes the amount of routing state information held at each level of the data center network.
- Automatically minimizes the amount of flooding.
- Automatic disaggregation of prefixes on link and node failures to prevent black-holing and suboptimal routing.
- Allows non-ECMP forwarding.
- Automatically rebalances traffic toward the spine based on the available bandwidth.
- Synchronizes a limited key-value data-store that can be used after protocol convergence; for example, to bootstrap higher levels of functionality on nodes.

For more details, see Internet draft draft-ietf-rift-rift-09 (expires May 7, 2020) *RIFT: Routing in Fat Trees*.

## Impact of Junos Implementation of RIFT Protocol on Network Performance

The integration of RIFT protocol into Junos OS has some impact on the route load and memory utilization for common datacenter architectures. This is because the RIFT protocol has the capability of consuming all available cores to improve protocol performance.

Dynamic configuration of RIFT is not fully supported. Configuration changes in the Junos OS CLI might restart the RIFT protocol causing the protocol to reconverge with resulting traffic loss.

## Unsupported Features with RIFT Protocol

The current Junos OS implementation of RIFT Internet draft draft-ietf-rift-rift-09 (expires May 7, 2020) *RIFT: Routing in Fat Trees* does not support:

- Logical systems
- SNMP
- In-service software upgrade (ISSU) and nonstop software upgrade
- Graceful Routing Engine switchover (GRES)
- Telemetry
- The current Junos OS implementation of RIFT does not implement:
  - Key-Value store
  - Horizontal links
  - Leaf-2-leaf support
  - Negative disaggregation
  - Mobility attributes on prefixes
  - Label binding on interfaces

## Enabling the RIFT Protocol

---

### SUMMARY



The RIFT protocol requires close to zero necessary configuration. When you enable the RIFT protocol, it automatically inherits the required configuration from the **junos-rift** package defaults, making IP fabrics simpler to manage.

---

The RIFT software package is a standalone package and the Juniper implementation of the protocol is executed in a modern memory and thread-safe programming language that is designed for optimal utilization of multi-core processor architectures.

The RIFT protocol initializes the associated RIFT processes, and the zero-touch configuration default values are applied through the configuration. It also automatically enables RIFT on all Ethernet interfaces. The **system-id** is automatically derived from the system MAC address, and the **level** is automatically determined through the discovery portion of the protocol operation.

## Before You Begin

You must download and install the RIFT software package on your device before you enable the protocol.

To install the RIFT protocol:

1. Download the special **junos-rift** package from the software package that is required to be run along with the baseline Junos OS software.

### NOTE:

- The baseline Junos OS software on which the **junos-rift** package is deployed must be 64-bit version only and starting from Junos OS Releases 19.4R1.
- While installing the **junos-rift** package, the devices must be cabled in a CLOS architecture.
- Because **junos-rift** is a separate package, it can be licensed separately from the Junos OS baseline package.

Licensing of the **junos-rift** package is granted under the [End User License Agreement](#) with special emphasis on sections 15, 16 and 17.

2. From the software package, extract the **junos-rift** package and download it to the **/var/tmp** directory on the host device.
3. After you successfully download the software package, run the following command:

```
user@host> request system software add package-name
```

For example:

```
user@host> request system software add /var/tmp/junos-rift.tgz
```

To enable the RIFT protocol, you must activate the RIFT software package on your device. You can activate RIFT either using the **request rift package activate** command, or manually load and combine the RIFT configuration from a specified file with the current configuration in the CLI.

### Enabling RIFT Using Activate Command

To activate the RIFT software package using the **request rift package activate** command:

1. After you successfully install the **junos-rift** package, run the following command:

```
user@host >request rift package activate
```

The activate command automatically commits the RIFT configuration.

### Enabling RIFT Using Load Command

To load the RIFT configuration manually:

1. After you successfully install the **junos-rift** package, run the following command to load and combine the RIFT configuration from a specified file with the current configuration in the CLI:

```
user@host# load merge /etc/config/junos-rift/package-defaults.conf
user@host# load merge /etc/config/junos-rift/platform/platform-defaults.conf
```

Here, *platform* is the host device, and can be any one of the following values—*mx*, *qfx*, or *vmx*.

2. Commit the configuration.

```
user@host> commit
```

### Enabling RIFT in CLOS-based Topology (ZTP Mode)

**NOTE:** For activating the RIFT software package on a CLOS topology, additional configuration is required. You must identify the nodes that are the top-of-fabric in the topology, and configure all the top-of-fabric devices to override the **auto** level in the default configuration.

To activate the RIFT software package in a CLOS-based topology:

1. Override the auto level in the default configuration, and optionally, specify the levels manually.

```
[edit protocols rift]
user@host# set level top-of-fabric
```

2. Commit the additional configuration.

```
user@host> commit
```

### Traceoptions for RIFT

Although enabling the RIFT protocol automatically inherits the necessary configuration, you can additionally configure minimal tracing as optional configuration.

To configure traceoptions for RIFT:

1. Specify the traceoptions and proxy-process parameters under the **rift** statement.

```
[edit protocols rift]
user@host# set traceoptions file size size
user@host# set traceoptions file files number
user@host# set traceoptions level level
user@host# set traceoptions flag flag
user@host# set proxy-process traceoptions file size size
user@host# set proxy-process traceoptions level level
user@host# set proxy-process traceoptions file files number
user@host# set proxy-process traceoptions flag flag
```

For example:

```
[edit protocols rift]
user@host# set traceoptions file size 1000000
user@host# set traceoptions file files 4
user@host# set traceoptions level info
user@host# set traceoptions flag node
```

```

user@host# set proxy-process traceoptions file size 1000000
user@host# set proxy-process traceoptions level info
user@host# set proxy-process traceoptions file files 4
user@host# set proxy-process traceoptions flag if-events

```

## Verifying RIFT Configuration

You can verify the RIFT protocol configuration from the following hierarchy levels:

- [groups rift-defaults]
- [interfaces interface-range rift-interfaces]
- [protocols rift]

```

[edit]
user@host# show groups rift-defaults
protocols {
  rift {
    node-id auto;
    level auto;
    lie-receive-address {
      family {
        inet 224.0.0.120;
        inet6 ff02::a1f7;
      }
    }
    interface <*> {
      lie-transmit-address {
        family {
          inet 224.0.0.120;
          inet6 ff02::a1f7;
        }
      }
      bfd-liveness-detection minimum-interval 1000;
    }
  }
}

```

```

[edit]
user@host# show interfaces interface-range rift-interfaces
member ge-0/0/*;
description "Match interfaces that RIFT could use.";

```

```
[edit]
user@host# show protocols rift
apply-groups rift-defaults;
interface rift-interfaces;
```

You can also verify the RIFT configuration by viewing the defaults applied from the **junos-rift** package. To do this, run the **show configuration protocols rift | display inherited** command.

For example:

```
user@host> show configuration protocols rift | display inherited
##
## 'auto' was inherited from group 'rift-defaults'
##
node-id auto;
level auto;
##
## 'lie-receive-address' was inherited from group 'rift-defaults'
##
lie-receive-address {
  ##
  ## 'family' was inherited from group 'rift-defaults'
  ##
  family {
    ##
    ## '224.0.0.120' was inherited from group 'rift-defaults'
    ##
    inet 224.0.0.120;
    ##
    ## 'ff02::alf7' was inherited from group 'rift-defaults'
    ##
    inet6 ff02::alf7;
  }
}
interface ge-0/0/0.1 {
  ##
  ## 'lie-transmit-address' was inherited from group 'rift-defaults'
  ##
  lie-transmit-address {
    ##
    ## 'family' was inherited from group 'rift-defaults'
    ##
    family {
      ##
```

```

## '224.0.0.120' was inherited from group 'rift-defaults'
##
inet 224.0.0.120;
##
## 'ff02::alf7' was inherited from group 'rift-defaults'
##
inet6 ff02::alf7;
}
}
##
## 'bfd-liveness-detection' was inherited from group 'rift-defaults'
## '400' was inherited from group 'rift-defaults'
##
bfd-liveness-detection minimum-interval 400;
}

```

[Table 3 on page 23](#) list the commands you can use to verify the RIFT protocol configuration and status.

**Table 3: Commands to Verify RIFT Protocol Configuration**

Command	Description
<b>show rift</b>	Inspect the runtime state of the RIFT protocol.
<b>show route protocol rift</b>	The RIFT protocol can be used in policies and commands where other protocols are accepted.
<b>show route protocol rift extensive display-client-data</b>	View detailed RIFT-installed routes.
<b>clear rift database content</b>	Clear the RIFT database.
<b>restart rift-proxyd</b>	Restart the RIFT protocol.

For more information, see the FAQ file in the software package distribution.

### Troubleshooting the RIFT Protocol

The RIFT protocol does not produce core files except in very extreme cases. It reports every failure by extensive logging and sometimes backtraces on exit. The RIFT process provides configurable tracing events that can be collected using traceoptions configuration.

To troubleshoot the RIFT implementation, see:

- **Forming Adjacency**

**Problem**

RIFT adjacency flapping up and down, showing rejects with **Multiple Neighbors** or **Remote Uses Our Own SystemID** errors.

**Solution**

The RIFT protocol does not support more than two neighbors on an Ethernet link forming a point-to-point adjacency, or a node's own interfaces looped back. Check and correct the cabling.

- **Undefined Level**

**Problem**

All the switches show undefined level and do not form three-way adjacencies, but link information elements (LIEs) are being sent and received.

**Solution**

There is a possibility that there is no top-of-fabric level configuration. All top-of-fabric devices must be configured with the top-of-fabric level to provide an anchor for ZTP.

- **Loopback Address**

**Problem**

Not able to get loopback addresses to my nodes in RIFT.

**Solution**

To have all the loopback addresses in top-of-fabric, the simplest way is to configure a loopback address in every node necessary and redistribute it in the northbound direction into RIFT. The following configuration is required for doing this:

```
[edit policy-options]
policy-statement lo0-rift {
  term 0 {
    from {
      protocol direct;
      route-filter loopback-address exact;
    }
    then accept;
  }
  term default {
    then reject;
  }
}
```

```
[edit protocols rift]
export {
  northbound {
    lo0-rift;
  }
}
```

This configuration allows every leaf to ping the loopback address of all other nodes except the top-of-fabric devices.

If the top-of-fabric devices should also be reachable from the leaf devices, or vice-versa, the top-of-fabric loopback addresses need to be exposed to one level below (that is, southbound). The following configuration is required for doing this:

```
[edit protocols rift]
export {
  southbound {
    lo0-rift;
  }
}
```

**NOTE:** To enable the top-of-fabric addresses to be propagated to all the leaf nodes, configure the **allow-rift-routes** option under the **[edit protocols rift export southbound]** hierarchy level.

#### • System Log Error Messages

The RIFT process generates system log messages to record errors related to the integration of the RIFT protocol into Junos OS. To interpret system log messages, refer to the following:

- **RIFT\_PROXYD\_ALREADY\_RUNNING**—Another instance of the RIFT process is already running.
- **RIFT\_PROXYD\_CONNECT\_RIFT**—Attempts to connect to the local RIFT process failed.

For more information on system log error messages, see [System Log Explorer](#).



# 2

CHAPTER

## Configuration Statements

---

capabilities (Protocols RIFT) | 29

default-prefixes (Protocols RIFT) | 30

export (Protocols RIFT) | 32

interface (Protocols RIFT) | 34

level (Protocols RIFT) | 37

lie-receive-address (Protocols RIFT) | 38

lie-transmit-address (Protocols RIFT) | 39

rft | 40

---



# capabilities (Protocols RIFT)

## Syntax

```
capabilities {
  (flood-reduction | no-flood-reduction);
}
```

## Hierarchy Level

```
[edit protocols rift]
```

## Release Information

Statement introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Enable the RIFT capability of reducing flooding globally.

If this capability is enabled, the node refloods TIEs only if it is the flood leader on the incoming link.

If you do not specify any of the options under the **capabilities** statement, the default behavior is to enable flood reduction.

## Options

**flood-reduction**—(Optional) Enable flood reduction globally.

**no-flood-reduction**—(Optional) Disable flood reduction globally.

**Default:** Flood reduction is enabled.

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## RELATED DOCUMENTATION

[rift | 40](#)

[Understanding Junos Implementation of Routing in Fat Tree \(RIFT\) Protocol | 15](#)

# default-prefixes (Protocols RIFT)

## Syntax

```
default-prefixes {
  family {
    (inet sets-of-ipv4-address | inet6 sets-of-ipv6-address);
  }
}
```

## Hierarchy Level

[edit protocols rift]

## Release Information

Statement introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Explicitly specify default prefixes generated in southbound direction.

This is helpful in cases where you have an internet default route and you want to inject the route from a leaf node without overlays. If the IP fabric originates default routes, you cannot differentiate between them and this might cause a blackhole. However, if you have all the internal fabric default routes on well known address prefixes, you can originate them and continue to use internet defaults in the IP fabric.

## Options

**family**—Specify the family type for the default prefixes to be generated in the southbound direction.

**inet *ipv4-address***—Specify the IPv4 IP addresses for the default prefixes to be generated in the southbound direction.

**inet6 *ipv6-address***—Specify the IPv6 IP addresses for the default prefixes to be generated in the southbound direction.

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## RELATED DOCUMENTATION



# export (Protocols RIFT)

## Syntax

```
export {  
  northbound {  
    policy;  
  }  
  southbound {  
    policy;  
    allow-rift-routes;  
  }  
}
```

## Hierarchy Level

[edit protocols rift]

## Release Information

Statement introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Enable exporting of routes from other protocols into the RIFT protocol.

## Options

**northbound**—(Optional) Policy to advertise routes from other protocols using north external prefix topology information elements (TIEs).

**southbound**—(Optional) Policy to advertise routes from other protocols using south external prefix TIEs.

**policy**—Name of the policy to be used for northbound or southbound exporting of routes into the RIFT protocol.

**allow-rift-routes**—(Optional) (Southbound Only) Allow calculated RIFT northbound routes in redistribution.

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## RELATED DOCUMENTATION

rift | 40

Understanding Junos Implementation of Routing in Fat Tree (RIFT) Protocol | 15

# interface (Protocols RIFT)

## Syntax

```
interface interface-name {
  allowed-authentication-keys (value | [set of values]);
  bfd-liveness-detection {
    minimum-interval milliseconds;
    multiplier milliseconds;
  }
  (check-common-instance-name | no-check-instance-name);
  (check-common-subnet | no-check-common-subnet);
  disable;
  lie-authentication;
  lie-origination-key;
  lie-transmit-address;
  lie-transmit-port port-number;
  metric metric;
  mode (active | advertise-subnets);
  (relax-three-way-nonce-check | no-relax-three-way-nonce-check);
  tie-receive-port port-number;
}
```

## Hierarchy Level

[edit protocols rift]

## Release Information

Statement introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Configure the interfaces for the RIFT protocol.

## Options

***interface-name***—Name of the interface on which the RIFT protocol should be configured.

**allowed-authentication-keys (value | [set of values])**—(Optional) Specify a single or set of values for allowed interface authentication keys (outer keys).

This allows you to set a set of key IDs that are allowed on this interface as outer security keys.

**Range:** 1 through 255



**bfd-liveness-detection**—(Optional) Configure Bidirectional Forwarding Detection (BFD) options. The BFD session is automatically brought up if it is configured on both sides of the session.

**Values:**

**minimum-interval** *milliseconds*—Specify the minimum transmit and receive interval.

**Range:** 1 through 255,000

**multiplier** *milliseconds*—Specify the detection time multiplier.

**Range:** 1 through 255

**check-common-instance-name**—(Optional) Enable check for common instance name advertised by neighboring device.

When multiple routing-instances of RIFT are running using the **routing-instance** statement, then enabling the **check-common-instance-name** option prevents forming mistaken adjacencies across different routing instances. The RIFT protocol declares links that receive a LIE with mismatched instance name as miscabled.

**check-common-subnet**—(Optional) Enable check for a common subnet on the neighboring device.

**disable**—(Optional) Disable the RIFT protocol on the specified interface.

**lie-authentication** (**loose** | **none** | **permissive** | **strict**)—(Optional) Specify the method to authenticate received LIEs (outer fingerprint).

**Values:**

**loose**—Verify authentication only if present, that is, when the key ID is not 0.

**none**—Disable authentication checking completely.

**permissive**—Accept authentication if key identifier is unknown.

**strict**—Accept authentication only if a key is present and it is valid.

**lie-origination-key** **lie-origination-key**—(Optional) Configure the key ID used to protect sent LIEs (outer key). You can configure to set the key used to authenticate LIEs, if required.

**Range:** 1 through 255

**lie-transmit-address**—(Optional) Configure the IPv4 or IPv6 IP address on which the link information elements (LIEs) should be sent. See [lie-transmit-address](#) for more information.

**lie-transmit-port** *port-number*—(Optional) Port on which the link information elements (LIEs) should be transmitted.

**Range:** 512 through 65535

**metric**—(Optional) Specify the advertised cost of the RIFT protocol interface.

**Range:** 1 through 134217727

**mode**—(Optional) Specify the mode of RIFT protocol interface.

**Values:**

- **active**—Run the RIFT protocol without advertising the Gigabit Ethernet interface subnets.
- **advertise-subnets**—Run the RIFT protocol and advertise the Gigabit Ethernet interface subnets.

**no-check-common-instance-name**—(Optional) Disable check for common instance name advertised by neighboring device.

**no-check-common-subnet**—(Optional) Disable check for a common subnet on the neighboring device.

**no-relax-three-way-nonce-check**—(Optional) Reject LIEs with undefined remote and local nonce in three-way.

**relax-three-way-nonce-check**—(Optional) Accept LIEs with undefined remote and local nonce in three-way.

This allows relaxation of the specification to accept undefined nonces in three-way state that allows for faster link bring-up after failures, but opens a security attack possibility (resetting adjacencies through replays).

**NOTE:** For maximum performance, the **relax-three-way-nonce-check** option should be on.

For maximum security when **lie-origination-key** is used, the **relax-three-way-nonce-check** option should be on.

**tie-receive-port *port-number***—(Optional) Port on which the topology information elements (TIEs) should be received.

**Range:** 512 through 65535

**Required Privilege Level**

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

**RELATED DOCUMENTATION**

[rift | 40](#)

[Understanding Junos Implementation of Routing in Fat Tree \(RIFT\) Protocol | 15](#)

# level (Protocols RIFT)

## Syntax

```
level {
  auto;
  configured-value value;
  leaf;
  top-of-fabric;
}
```

## Hierarchy Level

[edit protocols rift]

## Release Information

Statement introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Configure the level to identify the role of node in the RIFT topology.

If you do not specify any options under the **level** statement, the default option is **auto** level.

## Options

**auto**—(Optional) Enable zero touch provisioning to identify the level of the node automatically.

**configured-value *value***—(Optional) Enter the configured value of the level.

**Range:** 1 through 23

**leaf**—(Optional) Identify node as leaf in the RIFT topology. Automatically, this is level 0.

**top-of-fabric**—(Optional) Identify node as top-of-fabric in the RIFT topology. In automatic zero touch provisioning, this is level 24.

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## RELATED DOCUMENTATION

## lie-receive-address (Protocols RIFT)

### Syntax

```
lie-receive-address {  
  family {  
    (inet ipv4-address | inet6 ipv6-address);  
  }  
}
```

### Hierarchy Level

[edit protocols rift]

### Release Information

Statement introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

### Description

Configure the address on which the link information elements (LIEs) should be received.

### Options

**family**—Specify the family type for the RIFT protocol interface on which the LIEs should be received.

**inet *ipv4-address***—Specify the IPv4 IP address of the RIFT protocol interface on which the LIEs should be received.

**inet6 *ipv6-address***—Specify the IPv6 IP address of the RIFT protocol interface on which the LIEs should be received.

### Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

### RELATED DOCUMENTATION

[rift](#) | 40

Understanding Junos Implementation of Routing in Fat Tree (RIFT) Protocol | 15

# lie-transmit-address (Protocols RIFT)

## Syntax

```
lie-transmit-address {
  family {
    (inet ipv4-address | inet6 ipv6-address);
  }
}
```

## Hierarchy Level

```
[edit protocols rift interface interface-name]
```

## Release Information

Statement introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Configure the IPv4 or IPv6 IP address on which the link information elements (LIEs) should be sent to discover neighbors on the other side of the link.

## Options

**family**—Specify the family type for the RIFT protocol interface on which the LIEs should be sent.

**inet *ipv4-address***—Specify the IPv4 IP address of the RIFT protocol interface on which the LIEs should be sent.

**inet6 *ipv6-address***—Specify the IPv6 IP address of the RIFT protocol interface on which the LIEs should be sent.

## Required Privilege Level

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

## RELATED DOCUMENTATION

[interface](#) | 34

[rift](#) | 40

[Understanding Junos Implementation of Routing in Fat Tree \(RIFT\) Protocol](#) | 15

# rift

## Syntax

```
rift {
  capabilities;
  default-prefixes;
  default-prefixes-advertisement;
  export;
  external-preference external-preference;
  interface interface-name;
  level;
  lie-receive-address;
  lie-receive-port port-number;
  name name;
  node-id (node-id | auto);
  overload timeout seconds;
  preference route-preference;
  proxy-process traceoptions (file | flag | no-remote-trace);
  tie-authentication (loose | none | permissive | strict);
  tie-origination-key tie-origination-key;
  traceoptions (file | flag | level | peer-prefixes);
}
```

## Hierarchy Level

[edit protocols]

## Release Information

Statement introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Configure the Routing in Fat Trees (RIFT) protocol for IP fabrics that have CLOS-based and fat tree model topologies.

The RIFT protocol enables:

- Automatic disaggregation of prefixes on link and node failures.
- Minimal storing of routing state information at every level.
- Zero-configuration capabilities with automatic pruning.
- Load balancing of traffic towards the spine based on available bandwidth.

- Synchronization of a limited key-value data-store that can be used after protocol convergence.

## Options

**capabilities**—Enable the RIFT capability of reducing flooding globally. See [capabilities](#) for more information.

**default-prefixes**—Explicitly specify default prefixes generated in southbound direction. See [default-prefixes](#) for more information.

**default-prefixes-advertisement (always | never)**—(Optional) Enable default route generation strategy.

- **always**—Always originate default prefixes southbound.
- **never**—Never originate default prefix southbound.

**export**—Enable exporting of routes from other protocols into the RIFT protocol. See [export](#) for more information.

**external-preference *external-preference***—(Optional) External route preference for the RIFT protocol. The external preference configured is analogous to the OSPF configuration. This value is shown when RIFT installs the preference as static route values.

**Range:** 1 through 256

**interface**—(Optional) Configure the interfaces for the RIFT protocol. See [interface](#) for more information.

**level**—(Optional) Configure the level to identify the role of node in the RIFT topology. See [level](#) for more information.

**lie-receive-address**—(Optional) Configure the address on which the link information elements (LIEs) should be received. See [lie-receive-address](#) for more information.

**lie-receive-port *port-number***—(Optional) Port on which link information elements (LIEs) should be received.

**Range:** 512 through 65535

**name *name***—(Optional) Name of the node for identification.

**node-id (*node-id* | auto)**—(Optional) ID of the configured node, or enable zero touch provisioning where the node ID is configured automatically.

**overload**—(Optional) Configure the overload bit that takes the node out of traffic bearing paths.

### Values:

**timeout seconds**—(Optional) Specify in seconds the time after which the overload bit is reset. The overload timeout is similar to IS-IS timeout.

**Range:** 10 through 1800 seconds

**preference *route-preference***—(Optional) Route preference for the RIFT protocol.

**Range:** 15 through 256

**proxy-process**—(Optional) Configure the proxy process options for the RIFT protocol.



**tie-authentication** (**loose** | **none** | **permissive** | **strict**)—(Optional) Configure the method to authenticate received TIEs (inner fingerprint).

**Values:**

**loose**—Verify authentication only if present, that is, when the key ID is not 0.

**none**—Disable authentication checking completely.

**permissive**—Accept authentication if key identifier is unknown.

**strict**—Accept authentication only if a key is present and it is valid.

**tie-origination-key** *tie-origination-key*—(Optional) Configure the key ID used to protect self-originated TIEs (inner key).

**Range:** 1 through 16777215

**traceoptions**—(Optional) Enable traceoptions for the RIFT protocol.

**Required Privilege Level**

routing—To view this statement in the configuration.

routing-control—To add this statement to the configuration.

**RELATED DOCUMENTATION**

| [Understanding Junos Implementation of Routing in Fat Tree \(RIFT\) Protocol](#) | 15

# 3

CHAPTER

## Operational Commands

---

[show rift database](#) | **47**

[show rift flood-reduction](#) | **51**

[show rift interface](#) | **53**

[show rift node](#) | **59**

[show rift path-computation](#) | **62**

[show rift routes](#) | **64**

[show rift tie](#) | **68**

[show rift topology](#) | **70**

[show rift versions](#) | **73**

---



# show rift database

## Syntax

```
show rift database
content
statistics
```

## Release Information

Command introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Display Routing in Fat Trees (RIFT) protocol link-state database information. You can use this information for debugging purpose.

## Options

**content**—Display RIFT link-state database topology information element (TIE) headers information.

**statistics**—Display RIFT link-state database information.

## Required Privilege Level

view

## RELATED DOCUMENTATION

[rift | 40](#)

[Understanding Junos Implementation of Routing in Fat Tree \(RIFT\) Protocol | 15](#)

## List of Sample Output

[show rift database content on page 49](#)

[show rift database statistics on page 49](#)

## Output Fields

[Table 4 on page 47](#) lists the output fields for the **show rift database content** command. Output fields are listed in the approximate order in which they appear.

Table 4: show rift database content Output Fields

Field Name	Field Description
Dir	Direction of Topology information element (TIE).

Table 4: show rift database content Output Fields (*continued*)

Field Name	Field Description
Originator	Originating node.
Type	Type of TIE.
ID	TIE identifier.
SeqNr	TIE sequence number.
Lifetime	Remaining TIE lifetime in seconds.
Origin Creation Time	Time of TIE creation at the origin in seconds.
Origin Lifetime	Lifetime of TIE when created at the origin.
Content Size	Size of TIE as exchanged on the link.
Key ID	Displays whether the TIE has an inner security key, that is, carries originator's authentication.

Table 5 on page 48 lists the output fields for the **show rift database statistics** command. Output fields are listed in the approximate order in which they appear.

Table 5: show rift database statistics Output Fields

Field Name	Field Description
Peers	Configured peers.
in 3-WAY	Number of peers in three way.
Last UP/DOWN	Date and time peer was last UP or DOWN.
Last New TIE	Identification of last new TIE.
on	Date and time of the last new TIE.
TIE Version Collisions	Number of TIEs database received of same or older version.
Last Southbound Routes	Last time southbound routes were computed .
Dir	Direction of TIE.

Table 5: show rift database statistics Output Fields (*continued*)

Field Name	Field Description
Type	Type of TIE.
#TIES	Number of TIEs.

## Sample Output

**show rift database content**

user@host> **show rift database content**

```

Dir Originator Type ID SeqNr Lifetime Origin Creation Time
Origin Content Key ID
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
S 00000000000000001 Node 10000000 2829 601200 2019/11/08 22:28:42
604800 805 0
S 00000000000000001 Node 10000001 21020 601200 2019/11/08 22:28:42
604800 998 0
S 00000000000000001 Node 10000002 2340 601200 2019/11/08 22:28:42
604800 862 0
S 00000000000000001 Node 10000003 12074 601200 2019/11/08 22:28:42
604800 862 0

```

## Sample Output

**show rift database statistics**

user@host> **show rift database statistics**

```

Peers: Configured 4, in 3-WAY 4, Last UP/DOWN 2019/06/06 16:04:36.354
Last New TIE: 00002c6bf5586fc0/S/Positive/30000008, on 2019/06/06 16:08:56.187
TIE Version Collisions: 28
Last Southbound Routes: 2019/06/06 16:02:31.407

Dir   Type      #TIES
-----+-----+-----+
South

```

External	3
Node	6
Prefix	2
Positive	1

# show rift flood-reduction

## Syntax

```
show rift flood-reduction
statistics
```

## Release Information

Command introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Display Routing in Fat Trees (RIFT) protocol flood-reduction information.

## Options

**statistics**—Display RIFT flood-reduction statistics.

## Required Privilege Level

view

## RELATED DOCUMENTATION

<a href="#">rift   40</a>
<a href="#">Understanding Junos Implementation of Routing in Fat Tree (RIFT) Protocol   15</a>

## List of Sample Output

[show rift flood-reduction statistics on page 52](#)

## Output Fields

[Table 6 on page 51](#) lists the output fields for the **show rift flood-reduction statistics** command. Output fields are listed in the approximate order in which they appear.

Table 6: show rift flood-reduction statistics Output Fields

Field Name	Field Description
Runs	Number of runs.
Holds	Number of computations held down.
Leaders	Chosen flood leaders.
Last On	Date and time of last election.





# show rift interface

## Syntax

```
show rift interface
statistics
status
```

## Release Information

Command introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Display Routing in Fat Trees (RIFT) protocol interface information.

## Options

**statistics**—Display RIFT interface statistics.

**status**—Display status of RIFT interfaces.

## Required Privilege Level

view

## RELATED DOCUMENTATION

[rift | 40](#)

[Understanding Junos Implementation of Routing in Fat Tree \(RIFT\) Protocol | 15](#)

## List of Sample Output

[show rift interface on page 57](#)

[show rift interface status on page 58](#)

## Output Fields

[Table 7 on page 53](#) lists the output fields for the **show rift interface statistics** command. Output fields are listed in the approximate order in which they appear.

Table 7: show rift interface statistics Output Fields

Field Name	Field Description
Started	Date and time when the statistics was started.
TIES RX	Number of topology information elements (TIEs) received.

Table 7: show rift interface statistics Output Fields (*continued*)

Field Name	Field Description
TX	Number of TIEs transmitted.
REQ	Number of TIEs requested.
Neighbor REQ	Number of TIEs requested by neighbor.
TIDES TX	Number of topology information description elements (TIDES) transmitted.
TIRES TX	Number of topology information request elements (TIRES) transmitted.
TIRES RX	Number of TIRES received.
Pkt Rate/100msecs	Packet rate per 100 milliseconds: <ul style="list-style-type: none"> <li>• Highest—Highest achieved flooding rate.</li> <li>• Current—Current flooding rate n packets per 100 milliseconds.</li> <li>• Packet Sequence Losses—Sequence loses in flooding leading to flood rate adaptations.</li> </ul>
Last TIE RX	Last received TIE.
Last TIE RX on	Time at which last TIE was received.
Last Newer TIE RX	Last newer received TIE.
Last Newer TIE RX on	Time at which last newer TIE was received.
Last TIE TX	Last transmitted TIE.
Last TIE TX on	Time at which last TIE was transmitted.
Last Newer TIE RX	Last newer received TIE.
Last Newer TIE RX on	Time at which last newer TIE was received.
Last TIE TX	Last transmitted TIE.
Last TIE TX on	Time at which last TIE was transmitted.
TIE TX Queue Len	TIE transmission queue length.
Last TIE TX Queued	Last TIE queued for transmission.

Table 7: show rift interface statistics Output Fields (*continued*)

Field Name	Field Description
on	Event leading to queuing of last TIE.
Largest TX'ed - TIE/TIDE/TIRE	Largest transmitted size per packet type.
Three-Way UP	Number of times adjacency came up.
DOWN	Number of times adjacency came down.
Last UP	Date and time the last adjacency came up.
Last DOWN	Date and time the last adjacency went down.
Last Reason DOWN	Reason why the last adjacency went down.
LIE TX	Number of link information elements (LIEs) transmitted.
RX	Number of LIEs received.
Corrupt	Number of corrupt LIEs received.
Last LIE TX	Date and time the last LIE was transmitted.
Largest TX'ed	Size of largest transmitted TIE, TIDE, and TIRE.
Reject Reason	Reason for rejecting the last LIE.
Current Level Self/Neighbor	Current level of device (self) or the neighboring device.
Level Changes Self/Neighbor	Number of level changes for the device (self) or the neighboring device.
Flood Leader	Role as flood leader as designated by the neighboring device.
Changes	Number of changes in the flood relationship.
Last Change	Date and time the flood leadership changed last.

Table 8 on page 56 lists the output fields for the **show rift interface status** command. Output fields are listed in the approximate order in which they appear.

Table 8: show rift interface status Output Fields

Field Name	Field Description
Link ID	Link identifier of the local RIFT interface.
Interface	Interface name.
Status Admin	Administrative status of the interface. <ul style="list-style-type: none"> <li>• True—Status is up.</li> <li>• False—Status is down.</li> </ul>
Platform	Carrier status of interface. <ul style="list-style-type: none"> <li>• True—Status is up.</li> <li>• False—Status is down.</li> </ul>
BFD	BFD status of interface if BFD is negotiated. <ul style="list-style-type: none"> <li>• True—Status is up.</li> <li>• False—Status is down.</li> </ul>
State	State of interface: <ul style="list-style-type: none"> <li>• one-way</li> <li>• two-way</li> <li>• three-way</li> </ul>
Uptime	Duration interface was in three-way state.
LIE TX V4	Number of transmitted IPv4 LIEs.
LIE TX V6	Number of transmitted IPv6 LIEs.
LIE TX Port	Port on which LIEs are transmitted.
TIE RX Port	Port on which LIEs are received.
PoD	Point of delivery.
Nonce	Local security nonce.
Neighbor	If in three-way state, describes neighbor with HoldTime and Outer security key.
Link ID	Link ID of neighbor.

Table 8: show rift interface status Output Fields (*continued*)

Field Name	Field Description
Name	Link name of neighbor.
Level	Level of neighbor.
TIE V4	IPv4 addresses on which TIEs are received.
TIE V6	IPv6 addresses on which TIEs are received.
TIE Port	Port on which TIEs are received.
BW	Bandwidth of the interface.
Outer Key	Outer security key ID.
Holdtime	Adjacency hold time in 3-way.

## Sample Output

### show rift interface

user@host> show rift interface statistics

```

Link ID: 257, Interface: ge-0/0/1.0, Started: 2019/11/09 11:48:25.760
TIES RX: 65, TX: 6198, REQ: 1, Neighbor REQ: 2445
Same TIE RX: 5, TIES RE-TX: 714, Own TIES RX'ed: 6
TIDES TX: 104827, RX: 5442, TIRES TX: 33, TIRES RX: 1601
Pkt Rate/100msecs Highest: 50, Current: 50, Packet Sequence Losses: 22
Last TIE RX: 0000000000000002/S/Node____/10000003, Last TIE RX on: 2019/11/09
15:17:18.225
Last Newer TIE RX: 0000000000000002/S/Node____/10000000, Last Newer TIE RX on:
2019/11/09 15:06:54.486
Last TIE TX: 0000000000000001/S/PosExt___/700000b2, Last TIE TX on: 2019/11/09
15:17:23.747
TIE TX Queue Len: 0, Last TIE TX Queued: 0000000000000001/S/PosExt___/700001ff, on:

NewerOnTIDE
Last TIE REQ'ed: 0000000000000001/S/PosExt___/700000c5, Last TIE REQ on: 2019/11/09

15:07:00.033

```

```

Largest TX'ed: TIE: 1018, (0000000000000001/S/Node____/10000001), TIDE: 1192, TIRE:
    272
Three-Way UP 4, DOWN 3, Last UP 2019/11/09 13:54:22.950, Last DOWN 2019/11/09
13:54:19.777
Last Reason DOWN: HoldtimeExpired
LIE TX 6346, RX 6178
Last LIE TX 2019/11/09 15:19:54.824, RX 2019/11/09 15:19:53.715, Reject Reason:
None
Current Level Self 24, Neighbor 23, Level Changes Self 1, Neighbor 1
Flood Leader: False, Changes: 2, Last Change: 2019/11/09 11:48:26.049

```

## Sample Output

**show rift interface status**

```
user@host> show rift interface status
```

```

Link ID: 258, Interface: ge-0/0/0.1
Status Admin True, Platform True, BFD True, State: ThreeWay, 3-Way Uptime: 3 hours,
    34 minutes, 18 seconds
LIE TX V4: 224.0.0.120, LIE TX V6: ff02::alf7, LIE TX Port: 914, TIE RX Port: 915
PoD 0, Nonce 11589
Neighbor: ID 0000000000000000a, Link ID 258, Name: rift 10:ge-0/0/0.1, Level: 23
TIE V4: 1.1.10.2, TIE V6: fe80::5668:a300:114:2212, TIE Port: 915, BW 1000 Mbits/s
PoD: None, Nonce: 17251, Outer Key: 0, Holdtime: 3 secs

```

# show rift node

## Syntax

```
show rift node
statistics
status
```

## Release Information

Command introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Display Routing in Fat Trees (RIFT) protocol node information.

## Options

**statistics**—Display RIFT node statistics.

**status**—Display RIFT node status.

## Required Privilege Level

view

## RELATED DOCUMENTATION

- [rift | 40](#)
- [Understanding Junos Implementation of Routing in Fat Tree \(RIFT\) Protocol | 15](#)

## List of Sample Output

[show rift node statistics on page 61](#)

[show rift node status on page 61](#)

## Output Fields

[Table 9 on page 59](#) lists the output fields for the **show rift node statistics** command. Output fields are listed in the approximate order in which they appear.

Table 9: show rift node statistics Output Fields

Field Name	Field Description
Starttime	Time the statistics started.
Servicece Requests	Number of service requests.



Table 9: show rift node statistics Output Fields (*continued*)

Field Name	Field Description
Failed Requests	Number of failed requests.

Table 10 on page 60 lists the output fields for the **show rift node status** command. Output fields are listed in the approximate order in which they appear.

Table 10: show rift node status Output Fields

Field Name	Field Description
System Name	Name of the system.
System ID	Identification of the system.
Level	Level of the system.
RIFT Encoding Major	Major version of the RIFT protocol.
Minor	Minor version of the RIFT protocol.
Flags	Flags of the RIFT protocol.
Capabilities	Configured capabilities for the RIFT protocol: <ul style="list-style-type: none"> <li>• flood-reduction—True if enabled. False if disabled.</li> <li>• Hierarchy Indications—Indicates the hierarchy level.</li> </ul>
LIE v4 Receive	IPv4 receive multicast address of the link information element (LIE).
LIE v6 Receive	IPv6 receive multicast address of the LIE.
Re-Connections	Number of tries to connect to the Redis server.
Peers	Number of peers configured.
3-way	Number of peers in three way state.
South	Number of peers in 3-way southbound.
North	Number of peers in 3-way northbound.

## Sample Output

**show rift node statistics**

user@host> **show rift node statistics**

```
Starttime: 2019/06/06 16:01:00.786  
Service Requests: 46, Failed Requests: 0
```

## Sample Output

**show rift node status**

user@host> **show rift node status**

```
System Name: rift00, System ID: 00002c6bf5586fc0  
Level: 24, RIFT Encoding Major: 29, Minor: 0  
Flags: overload=False  
Capabilities: flood-reduction: True, Hierarchy Indications: top_of_fabric  
LIE v4 Receive: 224.0.0.120, LIE v6 Receive: ff02::alf7  
Re-Connections: 2278  
Peers: 4, 3-way: 4, South: 4, North: 0
```

# show rift path-computation

## Syntax

```
show rift path-computation
statistics
```

## Release Information

Command introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Display Routing in Fat Trees (RIFT) protocol path-computation information.

## Options

**statistics**—Display RIFT path computation statistics.

## Required Privilege Level

view

## RELATED DOCUMENTATION

<a href="#">rift   40</a>
<a href="#">Understanding Junos Implementation of Routing in Fat Tree (RIFT) Protocol   15</a>

## List of Sample Output

[show rift path-computation statistics on page 63](#)

## Output Fields

[Table 11 on page 62](#) lists the output fields for the **show rift path-computation statistics** command. Output fields are listed in the approximate order in which they appear.

Table 11: show rift path-computation statistics Output Fields

Field Name	Field Description
Dir	Direction of path computations.
Runs	Number of path computations.
Nodes#	Number of nodes after current path computation.

Table 11: show rift path-computation statistics Output Fields (*continued*)

Field Name	Field Description
Total	<ul style="list-style-type: none"> <li>• Nodes#—Total number of nodes in all path computations.</li> <li>• Prfxs#—Total prefixes in all path computations.</li> <li>• Deltas—Total generated prefix deltas in all path computations.</li> <li>• Holds—Number of times path computation was held down.</li> </ul>
Last Run	Last path computation.
Last Trigger	Last triggering TIE.
On	Last triggering TIE received on.

## Sample Output

### show rift path-computation statistics

user@host> show rift path-computation statistics

```

+----- Total -----+
Dir   Runs Nodes#|Nodes# Prfxs# Deltas Holds|      Last Run Last Trigger
              On
-----|-----
South  144      9  1227      0      0      0 16:08:56.187
00002c6bf5586fc0/S/Positive/30000008 16:08:56.187
North  144      1   142   2860   415   195 16:08:56.188
00002c6bf5c948c0/N/External/60000157 16:08:01.158

```

# show rift routes

## Syntax

```
show rift routes
content
next-hops
statistics
```

## Release Information

Command introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Display the routing table information of the Routing in Fat Trees (RIFT) protocol.

## Options

**content**—Display all route information of the RIFT protocol.

**next-hops**—Display next-hop information of the RIFT protocol.

**statistics**—Display the routing table statistics of the RIFT protocol.

## Required Privilege Level

view

## RELATED DOCUMENTATION

[rift | 40](#)

[Understanding Junos Implementation of Routing in Fat Tree \(RIFT\) Protocol | 15](#)

## List of Sample Output

[show rift routes content on page 66](#)

[show rift routes next-hops on page 66](#)

## Output Fields

[Table 12 on page 65](#) lists the output fields for the **show rift routes content** command. Output fields are listed in the approximate order in which they appear.

**Table 12: show rift routes content Output Fields**

Field Name	Field Description
Prefix	IPv4 or IPv6 prefix address.
Active	Type of active RIFT route.
Metric	Metric of next hop.
N-Hop	Next-hop ID.
All Present	Types of all present RIFT routes.

[Table 13 on page 65](#) lists the output fields for the **show rift routes next-hops** command. Output fields are listed in the approximate order in which they appear.

**Table 13: show rift routes next-hops Output Fields**

Field Name	Field Description
Nexthop	ID of RIFT next hop.
SystemID	System ID of the adjacent node.
Links	Link IDs leading to the adjacent node.

[Table 14 on page 65](#) lists the output fields for the **show rift routes statistics** command. Output fields are listed in the approximate order in which they appear.

**Table 14: show rift routes statistics Output Fields**

Field Name	Field Description
Nhops	Current number of next hops: <ul style="list-style-type: none"> <li>• Deletes—Number of next hops deleted.</li> <li>• Adds/Changes—Number of next hops added or changed.</li> </ul>
Last Transaction	Last transaction with: <ul style="list-style-type: none"> <li>• Nhop Diff#—Number of next hop differentials downloaded.</li> <li>• Route Diff#—Number of route differentials downloaded.</li> <li>• Route Type—Number of route types involved.</li> </ul>

Table 14: show rift routes statistics Output Fields (*continued*)

Field Name	Field Description
AF	<p>Numbers per address family for:</p> <ul style="list-style-type: none"> <li>• Prefixes—Number of prefixes in the table.</li> <li>• Deletes—Number of deletes performed over the lifetime.</li> <li>• Adds/Changes—Number of additions or changes performed over the lifetime.</li> </ul>

## Sample Output

**show rift routes content**

user@host> **show rift routes content**

Prefix	Active	Metric	N-Hop	All Present
-----+-----+-----+-----+-----				
1.24.66.0/30	N		2 80004f04	N
2.2.2.11/32	NExt		2 80004f19	NExt
2.2.2.12/32	NExt		2 80004f13	NExt
2.2.2.13/32	NExt		2 80004f03	NExt

**show rift routes next-hops**

user@host> **show rift routes next-hops**

Nexthop	SystemID	Links
-----+-----+-----		
80004f00	00002c6bf5a021c0	274 (ge-0/0/0.19)
80004f01	00002c6bf58703c0	270 (ge-0/0/0.15)
80004f02	00002c6bf5692fc0	278 (ge-0/0/0.23)
	00002c6bf55952c0	277 (ge-0/0/0.22)

## show rift routes statistics

user@host> **show rift routes statistics**

Nhops	Deletes	Adds/Changes
11	0	695

Last Transaction	Nhop Diff#	Route Diff#	Route Type
2019/05/31 10:44:43.594	0	4	NExt

AF	Prefixes	Deletes	Adds/Changes
IPv4	12	170	256
IPv6	9	423	496



# show rift tie

## Syntax

```
show rift tie tie
```

## Release Information

Command introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Display Routing in Fat Tress (RIFT) topology information element (TIE) information.

## Options

**tie**—Display information of the RIFT TIE in the following format:

```
(node-hex | node-name)/(North | South)/(node | prefix | positive | negative | key-value | external)/TIE-number-hex
```

Where:

- (node-hex | node-name)—System ID in hexadecimal number of node name.  
To specify the value in hexadecimal form, include 0x as a prefix.
- (North | South)—Choice of direction. You can use one letter abbreviation.
- (node | prefix | positive | negative | key-value | external)—Choice of type. You can use two letter abbreviations.
- TIE-number-hex—TIE number in hexadecimal.

To specify the value in hexadecimal form, include 0x as a prefix.

For example, `00002c6bf50f51c0/N/node/10000000`.

## Required Privilege Level

view

## RELATED DOCUMENTATION

[rift | 40](#)

[Understanding Junos Implementation of Routing in Fat Tree \(RIFT\) Protocol | 15](#)

## List of Sample Output

[show rift tie on page 69](#)

Output Fields

[Table 15 on page 69](#) lists the output fields for the **show rift tie** command. Output fields are listed in the approximate order in which they appear.

Table 15: show rift tie Output Fields

Field Name	Field Description
TIE ID	Identification of TIE.
Content	Content of TIE as python parsable structure conforming to RIFT encoding schemas.

Sample Output

**show rift tie**

user@host> **show rift tie 00002c6bf50f51c0/N/node/10000000**

```
TIE ID: 00002c6bf50f51c0/N/Node____/10000000
Content: TIEElement(node=NodeTIEElement(neighbors={48842194470336L:
NodeNeighborsTIEElement(bandwidth=1000, cost=1,
...
```

# show rift topology

## Syntax

```
show rift topology
nodes
```

## Release Information

Command introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Display the Routing in Fat Trees (RIFT) topology information.

## Options

**nodes**—Display information for all visible RIFT nodes.

## Required Privilege Level

view

## RELATED DOCUMENTATION

- [rift | 40](#)
- [Understanding Junos Implementation of Routing in Fat Tree \(RIFT\) Protocol | 15](#)

## List of Sample Output

[show rift topology nodes on page 71](#)

## Output Fields

[Table 16 on page 70](#) lists the output fields for the **show rift topology nodes** command. Output fields are listed in the approximate order in which they appear.

Table 16: show rift topology nodes Output Fields

Field Name	Field Description
Lvl	RIFT node level.
Name	RIFT node name.
Originator	Node system ID.

Table 16: show rift topology nodes Output Fields (*continued*)

Field Name	Field Description
Ovrlld	Node overloaded: <ul style="list-style-type: none"> <li>• Yes</li> <li>• No</li> </ul>
Dir	Direction of reachability. <ul style="list-style-type: none"> <li>• N—North direction.</li> <li>• S—South direction.</li> </ul>
3-way	Number of adjacencies in three way.
Miscbl	Number of miscabled links.
Secure	Number of secured links.
Auth	Number of authenticated links.
Non	Number of links without authentication.
V4	Number of originated IPv4 prefixes.
V6	Number of originated IPv6 prefixes.
Latest TIE Origination	Date of the newest TIE a node originated, that is, the last change it underwent.
Links	Number of secured (that is, authenticated through outer key) and unsecured links.
TIEs	Number of TIEs a node originated, both non-authenticated and secured by inner key.
Prefixs	Number of prefixes a node originated, both IPv4 and IPv6.

## Sample Output

```
show rift topology nodes
```

```
user@host> show rift topology nodes
```



# show rift versions

## Syntax

```
show rift versions
info
```

## Release Information

Command introduced in Junos OS Release 19.4R1 on MX Series routers, QFX Series switches, and vMX virtual routers.

## Description

Display various package version of the Routing in Fat Trees (RIFT) protocol.

## Options

**info**—Display RIFT package information.

## Required Privilege Level

view

## RELATED DOCUMENTATION

<a href="#">rift   40</a>
<a href="#">Understanding Junos Implementation of Routing in Fat Tree (RIFT) Protocol   15</a>

## List of Sample Output

[show rift versions info on page 74](#)

## Output Fields

[Table 17 on page 73](#) lists the output fields for the **show rift versions info** command. Output fields are listed in the approximate order in which they appear.

Table 17: show rift versions info Output Fields

Field Name	Field Description
Package	RIFT package version.
Build Date	Date and time of the RIFT package.
Encoding Version	RIFT protocol encoding version.

Table 17: show rift versions info Output Fields (*continued*)

Field Name	Field Description
Statistics Version	Statistics schema version.
Services Version	Services schema version.

## Sample Output

**show rift versions info**

user@host> **show rift versions info**

```
Package: 1.0.0.1064751
Built On: 2019-11-29T20:04:18.141027255+00:00
Encoding Version: 2.0
Statistics Version: 3.0
Services Version: 17.1
```