

IN FOCUS

Junos[®] OS Release 19.4

Published
2019-12-26

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

IN FOCUS Junos[®] OS Release 19.4

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

1

Start Here with Junos OS Release 19.4

What You Need to Know About the In Focus Guide | 7

Important Features in Junos OS Release 19.4 | 7

2

Inline Monitoring Services

How to Configure Inline Monitoring Services | 11

Understanding Inline Monitoring Services | 11

Benefits of Inline Monitoring Services | 11

Inline Monitoring Services Feature Overview | 12

Inline Monitoring Services Configuration Overview | 15

Supported and Unsupported Features with Inline Monitoring Services | 17

Configuring Inline Monitoring Services | 18

3

Junos Multi-Access User Plane

Junos Multi-Access User Plane | 25

Junos Multi-Access User Plane Overview | 26

Benefits of Junos Multi-Access User Plane | 28

4

Protect and Reroute Traffic Using BGP Labeled Unicast

How to Protect and Reroute Traffic Using BGP Labeled Unicast | 33

BGP PIC Edge Using BGP Labeled Unicast Overview | 33

Benefits of BGP PIC Edge Using BGP Labeled Unicast | 33

How does BGP Prefix Independent Convergence Work? | 34

BGP PIC Edge Using BGP Labeled Unicast as the Transport Protocol Overview | 34

5

Defining Flexible Algorithms for Segment Routing in IS-IS

How to Configure Flexible Algorithm in IS-IS for Segment Routing Traffic Engineering | 37

Understanding IS-IS Flexible Algorithm for Segment Routing | 37

Benefits of Configuring Flexible Algorithm | 38

What is Flexible Algorithm Definition (FAD)? | 38

Participation in a Flexible Algorithm	39
Network Topology Configured with Flexible Algorithm Definitions	40
Flexible Algorithm RIBs	44
BGP Community and Flexible Algorithms	44
Supported and Unsupported Features	44
Configuring Flexible Algorithm for Segment Routing Traffic Engineering	45

6

Routing in Fat Tree (RIFT)

How to Integrate RIFT protocol into Junos OS | 51

Understanding Junos Implementation of Routing in Fat Tree (RIFT) Protocol	51
Benefits of RIFT Protocol	51
RIFT Protocol Overview	52
Impact of Junos Implementation of RIFT Protocol on Network Performance	53
Unsupported Features with RIFT Protocol	53
Enabling the RIFT Protocol	53

7

Wi-Fi Mini-PIM

Wi-Fi Mini-PIM Overview | 65

Wi-Fi Mini-Physical Interface Module Overview	65
Features Supported on the Wi-Fi Mini-PIM	66
Benefits of Using the Wi-Fi Mini-PIM	67

1

CHAPTER

Start Here with Junos OS Release 19.4

[What You Need to Know About the In Focus Guide | 7](#)

[Important Features in Junos OS Release 19.4 | 7](#)

What You Need to Know About the In Focus Guide

Use this guide to quickly learn about important features in this Junos OS Release 19.4 and how you can deploy them in your network.

You might also be interested in seeing the complete list of features in the [Release Notes for Junos OS Release 19.4](#). In addition to this guide, you can find detailed information on concepts, configuration, and examples in the [Junos OS documentation](#).

Want to tell us what you think about this guide? E-mail us at techpubs-comments@juniper.net.

Important Features in Junos OS Release 19.4

For details on these features, go to the other chapters in this guide or click the link in the feature description below.

- **Inline monitoring services (MX Series with MPCs excluding MPC10E and MPC11E linecards)**—Starting in Junos OS Release 19.4R1, you can configure a new monitoring technology that provides the flexibility to monitor different streams of traffic at different sampling rates on the same interface. You can also export the packet up to the configured clip length to a collector in an IP Flow Information Export (IPFIX) format. The IPFIX format includes important metadata information about the monitored packets for further processing at the collector.

The inline monitoring services overcome the limitations of traditional sampling technologies, such as JFlow, sFlow, and port mirroring, thereby providing you the benefit of effective sampling and troubleshooting processes.

[See [“How to Configure Inline Monitoring Services”](#) on page 11.]

- **Integrating RIFT protocol into Junos OS (MX Series, QFX Series, and VMX virtual routers)**—Starting in Junos OS Release 19.4R1, you can integrate a new IGP protocol, Routing in Fat Tree (RIFT), into Junos OS to route packets in variants of CLOS-based and fat tree network topologies (also called the spine and leaf model).

The RIFT protocol is capable of automatic construction of fat-tree topologies, providing you the benefit of having a close to zero necessary configuration. RIFT makes networks resilient, extensively traceable, and simpler to manage, thereby overcoming the deployment limitations of evolving IP fabrics.

[See [“How to Integrate RIFT protocol into Junos OS”](#) on page 51.]

- **Junos Multi-Access User Plane (MX240, MX480, MX960)**—With Junos OS Release 19.4R1, we introduce Junos Multi-Access User Plane, a software solution that turns your MX router into a high-capacity user plane function called a System Architecture Evolution Gateway-User Plane (SAEGW-U). This MX

SAEGW-U interoperates with a third-party SAEGW-C (control plane function), per 3GPP Release 14 Control User Plane Separation (CUPS) architecture, to provide high-throughput 4G and 5G fixed-wireless access service with support for 5G non-stand-alone (NSA) mode. CUPS enables independent scaling of the user and control planes, network architecture flexibility, operational flexibility, and an easier migration path from 4G to 5G services. The CUPS architecture is optional for 4G but inherent in 5G architecture.

To transform your MX240, MX480, or MX960 router into an SAEGW-U, all you need is at least one MPC7 linecard, a routing engine with at least 32GB memory, and Junos OS Release 19.4R1.

[See [“Junos Multi-Access User Plane” on page 25.](#)]

- **Microsoft Azure Key Vault (HSM) integration (vSRX 3.0)**—Starting in Junos OS Release 19.4R1, vSRX 3.0 is integrated with Microsoft Azure Key Vault hardware security module (HSM). With the integration of Microsoft Azure Key vault HSM, vSRX can protect and manage sensitive data such as private cryptographic keys, passwords, and configurations.

[See [Deployment of Microsoft Hardware Security Module on vSRX 3.0.](#)]

- **Support for BGP PIC Edge with BGP labeled unicast (MX Series and PTX Series)**—Starting with Junos OS Release 19.4R1, MX Series and PTX Series routers support BGP PIC Edge with BGP labeled unicast as the transport protocol. BGP PIC Edge using the BGP labeled unicast transport protocol helps to protect and reroute traffic when border nodes (ABR and ASBR) failures happen in multi-domain networks. Multi-domain networks are typically used in Metro Ethernet aggregation and Mobile Backhaul networks designs.

[See [“How to Protect and Reroute Traffic Using BGP Labeled Unicast” on page 33.](#)]

- **Support for flexible algorithm in IS-IS for segment routing–traffic engineering (MX Series and PTX Series)**—Starting in Junos OS Release 19.4R1, you can thin slice a network by defining flexible algorithms that compute paths using different parameters and link constraints based on your requirements. For example, you can define a flexible algorithm that computes a path to minimize IGP metric and define another flexible algorithm to compute a path based on traffic engineering metric to divide the network into separate planes. This feature allows networks without a controller to configure traffic engineering and utilize segment routing capability of a device.

To define a flexible algorithm, include **flex-algorithm** statement at the **[edit routing-options]** hierarchy level.

To configure participation in a flexible algorithm include the **flex-algorithm** statement at the **[edit protocols isis segment routing]** hierarchy level.

[See [“How to Configure Flexible Algorithm in IS-IS for Segment Routing Traffic Engineering” on page 37.](#)]

- **Wi-Fi Mini-Physical Interface Module (SRX320, SRX340, SRX345, and SRX550M)**—In Junos OS Release 19.4R1, we introduce the Wi-Fi Mini-Physical Interface Module (Mini-PIM). For retail and small offices, the Wi-Fi Mini-PIM provides secure wireless LAN connectivity to endpoint devices. The Wi-Fi Mini-PIM supports 802.11ac wave 2 wireless standards.

[See [Wi-Fi Mini-Physical Interface Module Overview.](#)]

2

CHAPTER

Inline Monitoring Services

[How to Configure Inline Monitoring Services](#) | **11**

How to Configure Inline Monitoring Services

SUMMARY

Inline monitoring services enable you to monitor IPv4 and IPv6 packets at different sampling rates, and export the actual packet up to the configured clip length to a collector. The monitored packets are exported in an IP Flow Information Export (IPFIX) format, which includes important metadata information for further processing of the packets. With this feature, you can benefit from flexible monitoring and effective troubleshooting processes.

IN THIS SECTION

- [Understanding Inline Monitoring Services | 11](#)
- [Configuring Inline Monitoring Services | 18](#)

Understanding Inline Monitoring Services

IN THIS SECTION

- [Benefits of Inline Monitoring Services | 11](#)
- [Inline Monitoring Services Feature Overview | 12](#)
- [Inline Monitoring Services Configuration Overview | 15](#)
- [Supported and Unsupported Features with Inline Monitoring Services | 17](#)

Benefits of Inline Monitoring Services

Flexible—Inline monitoring services allow different inline-monitoring instances to be mapped to different firewall filter terms, unlike in traditional sampling technologies, where all the instances are mapped to the Flexible PIC Concentrator (FPC). This provides you with the flexibility of sampling different streams of traffic at different rates on a single interface.

Packet format agnostic—Traditional flow collection technologies rely on packet parsing and aggregation by the network element. With inline monitoring services, the packet header is exported to the collector

for further processing, but without aggregation. Thereby, you have the benefit of using arbitrary packet fields to process the monitored packets at the collector.

Inline Monitoring Services Feature Overview

Service providers and content providers typically require visibility into traffic flows in order to evaluate peering agreements, detect traffic anomalies and policy violations, and monitor network performance. To meet these requirements, you would traditionally export aggregate flow statistics information using Netflow, JFlow, or IPFIX variants.

As an alternative approach, you can have the packet content sampled, add metadata information, and export the monitored packets to an collector. The inline monitoring services enables you to do this on MX Series routers with MPCs excluding MPC10E and MPC11E linecards.

With inline monitoring services, you can monitor every IPv4 and IPv6 packet on both ingress and egress directions of an interface. Junos OS encapsulates the monitored traffic in an IPFIX format and exports the actual packet up to the configured clip length to an collector for further processing. By default, Junos OS supports a maximum clip length of 126 bytes starting from the Ethernet header.

Figure 1 on page 12 illustrates the IPFIX format specification.

Figure 1: Inline Monitoring IPFIX Specification

Ethernet	
IP	
UDP	
IPFIX Header	
Set	
Information Elements	

ID	Length	Description	Details
10	4B	ingressInterface	SNMP index of incoming interface
14	4B	egressInterface	SNMP index of outgoing interface when flowDirection=Output, otherwise 0.
61	1B	flowDirection	Direction (0: Input , 1:Output)
312	2B	dataLinkFrameSize	Length of sampled data link frame
315	Variable	dataLinkFrameSelection	N octet from data link frame of monitored packet. Reports actual monitored packet starting from Layer 2 as [Ethernet header/802.1Q header(any)/IP header/Payload ...] up to configured maximum-clip-length

The IPFIX header and IPFIX payload are encapsulated using IP or UDP transport layer. The exported IPFIX format includes two data records and two data templates that are exported to every collector:

- Data record—Includes incoming and outgoing interface, flow direction, data link frame section, and data link frame size. This information is sent to the collector only when sampled packets are being exported.

Figure 2 on page 14 is a sample illustration of IPFIX data record packet.

- Option data record—Includes system level information, such as exporting process ID, and sampling interval. This information is sent to the collector periodically, irrespective of whether sampling packets are being exported are not.

Figure 3 on page 14 is a sample illustration of IPFIX option data record packet.

Table 1: Information Element fields in IPFIX Option Data Packet

Number	Information Element ID	Information Element Length	Details
1	144	4B	Observation domain ID - An unique identifier of exporting process per IPFIX device. Purpose of this field is to limit the scope of other information element fields.
2	34	4B	Sampling interval at which the packets are sampled. 1000 indicates that one of 1000 packets is sampled.

- Data template—Includes five information elements:
 - Ingress interface
 - Egress interface
 - Flow direction
 - Data link frame size
 - Variable data link frame selection

Figure 4 on page 15 is a sample illustration of IPFIX data template packet.

- Option data template—Includes flow exporter and sampling interval information.

Figure 5 on page 15 is a sample illustration of IPFIX option data template packet.

When there is new or change in the inline monitoring services configuration, periodic export of data template and option data template is immediately sent to the respective collectors.

Figure 2: IPFIX Data Record

```

Version: 10
Length: 160
▶ Timestamp: Feb 28, 2019 14:05:41.000000000 IST
FlowSequence: 474
Observation Domain Id: 1342242816
▼ Set 1 [id=2000] (1 flows)
  FlowSet Id: (Data) (2000)
  FlowSet Length: 144
  [Template Frame: 91]
  ▼ Flow 1
    InputInt: 553
    OutputInt: 0
    Direction: Ingress (0)
    Data Link Frame Size: 1496
    ▼ Data Link Frame Section: 80711f7ce252000001000e000800450005ca000000004011...
      String_len_short: 128

```

Figure 3: IPFIX Option Data Record

```

Version: 10
Length: 28
▶ Timestamp: Feb 28, 2019 14:21:10.000000000 IST
FlowSequence: 11
Observation Domain Id: 1342242816
▼ Set 1 [id=2600] (1 flows)
  FlowSet Id: (Data) (2600)
  FlowSet Length: 12
  [Template Frame: 11]
  ▼ Flow 1
    FlowExporter: 1
    Sampling interval: 1

```

Figure 4: IPFIX Data Template

```

Version: 10
Length: 44
> Timestamp: Feb 28, 2019 14:05:42.000000000 IST
FlowSequence: 474
Observation Domain Id: 1342242816
▼ Set 1 [id=2] (Data Template): 2000
  FlowSet Id: Data Template (V10 [IPFIX]) (2)
  FlowSet Length: 28
  ▼ Template (Id = 2000, Count = 5)
    Template Id: 2000
    Field Count: 5
    ▼ Field (1/5): INPUT_SNMP
      0... .. = Pen provided: No
      .000 0000 0000 1010 = Type: INPUT_SNMP (10)
      Length: 4
    ▼ Field (2/5): OUTPUT_SNMP
      0... .. = Pen provided: No
      .000 0000 0000 1110 = Type: OUTPUT_SNMP (14)
      Length: 4
    ▼ Field (3/5): DIRECTION
      0... .. = Pen provided: No
      .000 0000 0011 1101 = Type: DIRECTION (61)
      Length: 1
    ▼ Field (4/5): dataLinkFrameSize
      0... .. = Pen provided: No
      .000 0001 0011 1000 = Type: dataLinkFrameSize (312)
      Length: 2
    ▼ Field (5/5): dataLinkFrameSection
      0... .. = Pen provided: No
      .000 0001 0011 1011 = Type: dataLinkFrameSection (315)
      Length: 65535 [i.e.: "Variable Length"]

```

Figure 5: IPFIX Option Data Template

```

Version: 10
Length: 36
> Timestamp: Feb 28, 2019 14:21:10.000000000 IST
FlowSequence: 11
Observation Domain Id: 1342242816
▼ Set 1 [id=3] (Options Template): 2600
  FlowSet Id: Options Template (V10 [IPFIX]) (3)
  FlowSet Length: 20
  ▼ Options Template (Id = 2600) (Scope Count = 1; Data Count = 1)
    Template Id: 2600
    Total Field Count: 2
    Scope Field Count: 1
    ▼ Field (1/1) [Scope]: FLOW_EXPORTER
      0... .. = Pen provided: No
      .000 0000 1001 0000 = Type: FLOW_EXPORTER (144)
      Length: 4
    ▼ Field (1/1): SAMPLING_INTERVAL
      0... .. = Pen provided: No
      .000 0000 0010 0010 = Type: SAMPLING_INTERVAL (34)
      Length: 4
    Padding: 0000

```

Inline Monitoring Services Configuration Overview

You can configure a maximum of sixteen inline-monitoring instances that support template and collector-specific configuration parameters. Each inline monitoring instance supports up to four collectors (maximum of 64 collectors in total), and you can specify different sampling rates under each collector

configuration. Because of this flexibility, the inline monitoring services overcome the limitations of traditional sampling technologies, such as JFlow, sFlow, and port mirroring.

To configure inline monitoring:

1. You must include the **inline-monitoring** statement at the **[edit services]** hierarchy level. Here you specify the template and inline monitoring instance parameters. You must specify the collector parameters under the inline-monitoring instance.
2. Specify arbitrary match conditions using a firewall filter term and an action to accept the configured inline-monitoring instance. This maps the inline-monitoring instance to the firewall term.
3. Map the firewall filter under the family **inet** or **inet6**. You can also alternatively apply the firewall filter to a forwarding table filter with input or output statement to filter ingress or egress packets, respectively.

Remember:

- The device must support a maximum packet length (clip length) of 126 bytes to enable inline monitoring services.
- You cannot configure more than 16 inline-monitoring instances because of the scarcity of bits available in the packet in the forwarding path.
- Apply inline monitoring services only on a collector interface, that is, the interface on which the collector is reachable. You must not apply inline monitoring on IPFIX traffic as this generates another IPFIX packet for sampling thereby creating a loop. This includes inline monitoring service-generated traffic, such as template and record packet, option template and option record packet.
- When inline monitoring service is enabled on aggregated Ethernet (AE) interfaces, the information element values are as follows:

Table 2: Information Element Values for Aggregated Ethernet Interfaces

Direction of inline monitoring service on AE interface	Information element-10 (Incoming interface)	Information element-14 (Outgoing interface)
Ingress	SNMP ID of AE	0
Egress	SNMP ID of AE	SNMP ID of member link

- When inline monitoring service is enabled on IRB interfaces, the information element values are as follows:

Table 3: Information Element Values for IRB Interfaces

Direction of inline monitoring service on IRB interface	Information element-10 (Incoming interface)	Information element-14 (Outgoing interface)

Table 3: Information Element Values for IRB Interfaces (*continued*)

Ingress	SNMP ID of IRB	0
Egress	SNMP ID of IRB	SNMP ID of vlan-bridge encapsulated interface

- For XL-XM based devices (with Lookup chip (XL) and buffering ASIC (XM)), the length of the Data Link Frame Section information element in an exported packet can be shorter than the clip length even if the egress packet length is greater than clip length.

The length of the Data Link Frame Section information element is reduced by 'N' number of bytes where 'N' = (ingress packet Layer 2 encapsulation length - egress packet Layer 2 encapsulation length).

For instance, the Layer 2 encapsulation length for the ingress packet is greater than that of the egress packet when the ingress packet has MPLS labels and egress packet is of IPv4 or IPv6 type. When traffic flows from the provider edge (PE) device to the customer edge (CE) device, the ingress packet has VLAN tags and the egress packet is untagged.

In such cases, the clip length can go past the last address location of the packet head, generating a **PKT_HEAD_SIZE** system log message. This can result in degradation of packet forwarding for the device.

- In case of inline monitoring services in the ingress direction, the **egressInterface** (information element ID 14) does not report SNMP index of the output interface. This information element ID always reports value zero in case of ingress direction. The receiving collector process should identify the validity of this field based on the **flowDirection** (information element ID 61).

Supported and Unsupported Features with Inline Monitoring Services

Inline monitoring services supports:

- Graceful Routing Engine switchover
- In-service software upgrade (ISSU), nonstop software upgrade (NSSU), and nonstop active routing (NSR)
- Ethernet interfaces and integrated routing and bridging (IRB) interfaces
- Junos node slicing

Inline monitoring services currently does not support:

- Ability to configure more than 16 inline-monitoring instances.
- Junos Traffic Vision
- Inline-monitoring-instance action is supported only for **inet** and **inet6** firewall filters. It is not supported for other family filters.
- IPv6 addressable collectors

- Virtual platforms
- Logical systems

Configuring Inline Monitoring Services

SUMMARY

You can configure inline monitoring services to monitor different streams of traffic at different sampling rates on the same logical unit of the interface. You can also export the original packet size to an collector along with information on the interface origin for effective troubleshooting.

The inline monitoring services can monitor both IPv4 and IPv6 traffic on both ingress and egress directions. You can enable inline monitoring on MX Series routers with MPCs excluding MPC10E and MPC11E linecards.

Before You Configure

When you configure inline monitoring services, you can:

- Configure up to 16 inline-monitoring instances. Under each instance, you can configure specific collector and template parameters.
- Configure up to 4 IPv4-addressable collectors under each inline-monitoring instance. In total, you can configure up to 64 collectors. The collectors can be remote, and at different locations.

For each collector, you can configure specific parameters, such as source, destination address, sampling rate, forwarding class, and so on. The default routing-instance name at the collector is **default.inet**.

- Configure **inet** or **inet6** family firewall filter with the term action **inline-monitoring-instance** *inline-monitoring-instance-name*.

Each term can support a different inline-monitoring instance.

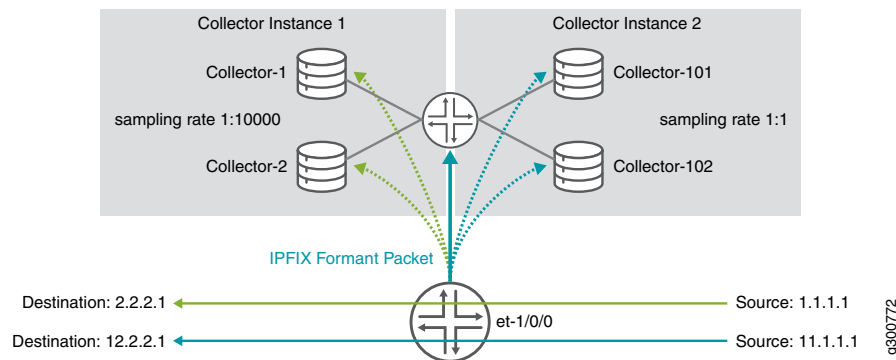
- Attach the inline monitoring firewall filter under **inet** or **inet6** family of the logical unit of the interface.

After successfully committing the configuration, you can verify the implementation of the inline monitoring services by issuing the *show services inline-monitoring statistics fpc-slot* command from the device CLI.

NOTE: If a packet requires inline monitoring services to be applied along with any of the traditional sampling technologies (such as, JFlow, SFlow, or port mirroring), the Packet Forwarding Engine performs both inline monitoring services and the traditional sampling technology on that packet.

Figure 6 on page 19 is a sample illustration of inline monitoring services, where traffic is monitored at two different sampling rates on the device interface, and exported to four remote collectors in an IPFIX encapsulation format.

Figure 6: Inline Monitoring Services



In this example, the et-1/0/0 interface of the device is configured with inline monitoring services. The details of the configurations are as follows:

- There are two inline-monitoring instances — Instance 1 and Instance 2.
- There are four collectors, two collectors under each inline monitoring instance.
 - Instance 1 has Collector-1 and Collector-2.
 - Instance 2 has Collector-101 and Collector-102.
- The collectors on Instance 1 have a sampling rate of 1:10000.
- The collectors on Instance 2 have a sampling rate of 1:1.
- Instance 1 collectors have a source and destination address of 1.1.1.1 and 2.2.2.1, respectively.
- Instance 2 collectors have a source and destination address of 11.1.1.1 and 12.2.2.1, respectively.
- The packets are exported to the collectors in an IPFIX encapsulated format.

To configure inline monitoring services:

1. Define a firewall filter for each inline-monitoring instance for servicing the inline monitoring services. You can configure **inet** or **inet6** family firewall filter with the term action **inline-monitoring-instance**.

To define a firewall filter:

```
[edit firewall famil family filter filter-name term term]
user@host# set from source-address source-IPv4-address
user@host# set from destination-address destination-IPv4-address
user@host# set then inline-monitoring-instance inline-monitoring-instance-name
user@host# set then action
```

In this example, Terms t1 and t2 are configured for Instance1 and Instance2, respectively.

```
[edit firewall famil inet filter SAMPLE_FOR_1 term t1]
user@host# set from source-address 1.1.1.0/24
user@host# set from destination-address 2.2.2.0/24
user@host# set then inline-monitoring-instance Instance1
user@host# set then accept
user@host# set term t2 from source-address 11.1.1.0/24
user@host# set term t2 from destination-address 12.2.2.0/24
user@host# set term t2 then inline-monitoring-instance Instance2
user@host# set term t2 then accept
```

2. Enable inline monitoring services by configuring the associated template, instance, and collector parameters.
 - a. To configure the inline monitoring services template:

```
[edit services inline-monitoring template template-name]
user@host# set template-refresh-rate template-refresh-rate
user@host# set option-template-refresh-rate option-template-refresh-rate
user@host# set observation-domain-id observation-domain-id
```

In this example, templates template-1 and template-2 are configured.

```
[edit services inline-monitoring template template-1]
user@host# set template-refresh-rate 60
user@host# set option-template-refresh-rate 100
user@host# set observation-domain-id 1
[edit services inline-monitoring template template-2]
user@host# set template-refresh-rate 60
user@host# set option-template-refresh-rate 100
user@host# set observation-domain-id 2
```

- b. To configure inline monitoring instance and collector parameters:

```
[edit services inline-monitoring instance inline-monitoring-instance-name]
user@host# set template-name template-name
user@host# set maximum-clip-length maximum-clip-length
user@host# set collector collector-name source-address source-IPv4-address
user@host# set collector collector-name destination-address destination-IPv4-address
user@host# set collector collector-name destination-port destination-port
user@host# set collector collector-name sampling-rate sampling-rate
```

In this example, Instance1 has two collectors, collector-1 and collector-2, and Instance2 has two collectors, collector-101 and collector-102. Different sampling rates have been configured for both the instances.

```
[edit services inline-monitoring instance Instance1]
user@host# set template-name template-1
user@host# set maximum-clip-length 126
user@host# set collector collector-1 source-address 1.1.1.1
user@host# set collector collector-1 destination-address 2.2.2.1
user@host# set collector collector-1 destination-port 2055
user@host# set collector collector-1 sampling-rate 10000
user@host# set collector collector-2 source-address 1.1.1.1
user@host# set collector collector-2 destination-address 2.2.2.1
user@host# set collector collector-2 destination-port 2055
user@host# set collector collector-2 sampling-rate 10000
```

```
[edit services inline-monitoring instance Instance2]
user@host# set template-name template-2
user@host# set maximum-clip-length 126
user@host# set collector collector-101 source-address 11.1.1.1
user@host# set collector collector-101 destination-address 12.2.2.1
user@host# set collector collector-101 destination-port 2055
user@host# set collector collector-101 sampling-rate 1
user@host# set collector collector-102 source-address 11.1.1.1
user@host# set collector collector-102 destination-address 2.2.2.1
user@host# set collector collector-102 destination-port 2055
user@host# set collector collector-102 sampling-rate 1
```

3. Map the firewall filter under the family **inet** or **inet6** of the logical unit of the interface to apply inline monitoring in the ingress or egress direction.

Alternatively, you can apply inline monitoring by mapping the firewall filter to a forwarding table filter with input or output statement to filter ingress or egress packets, respectively.

To attach the firewall filter:

```
[edit interfaces interface-name]  
user@host# set unit 0 family family filter input filter  
user@host# set unit 0 family family address ip-address
```

In this example, the inline monitoring filter is attached to family inet of unit 0 of et-1/0/0.

```
[edit interfaces et-1/0/0]  
user@host# set unit 0 family inet filter input SAMPLE_FOR_1  
user@host# set unit 0 family inet address 10.100.0.1/30
```

WHAT'S NEXT

For more information on configuring inline monitoring services, see the [Monitoring, Sampling, and Collection Services Interfaces Feature Guide](#).

3

CHAPTER

Junos Multi-Access User Plane

Junos Multi-Access User Plane | 25

Junos Multi-Access User Plane

SUMMARY

With Junos OS Release 19.4R1, we introduce Junos Multi-Access User Plane, a software solution that turns your MX router into a high-capacity user plane function called a System Architecture Evolution Gateway-User Plane (SAEGW-U). This MX SAEGW-U interoperates with a third-party SAEGW-C (control plane function), per 3GPP Release 14 Control User Plane Separation (CUPS) architecture, to provide high-throughput 4G and 5G fixed-wireless access service with support for 5G non-stand-alone (NSA) mode. CUPS enables independent scaling of the user and control planes, network architecture flexibility, operational flexibility, and an easier migration path from 4G to 5G services. The CUPS architecture is optional for 4G but inherent in 5G architecture.

IN THIS SECTION

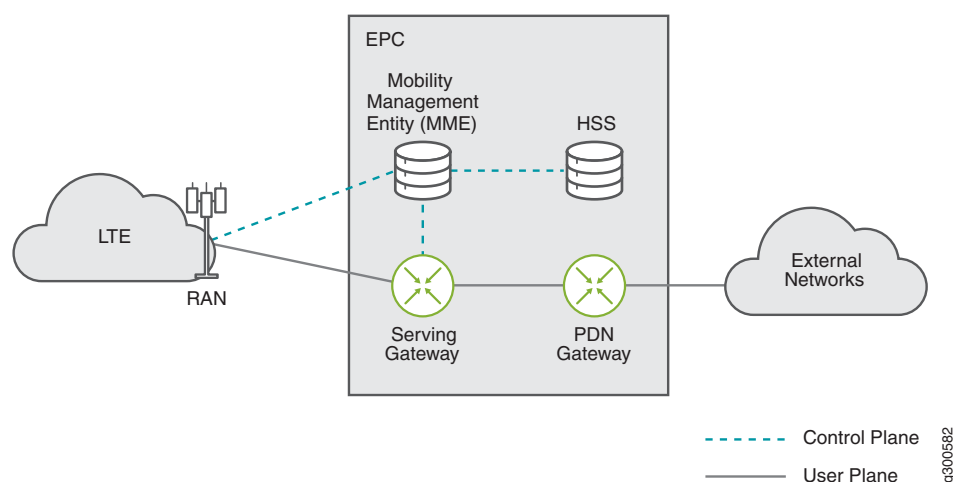
- [Junos Multi-Access User Plane Overview | 26](#)

Junos Multi-Access User Plane Overview

The 3GPP Release 8 introduced the Evolved Packet Core (EPC) for core network architecture. As [Figure 7 on page 26](#) shows, the four main EPC network elements are:

- Serving Gateway
- Packet Data Network (PDN) Gateway
- Mobility Management Entity
- Home Subscriber Server

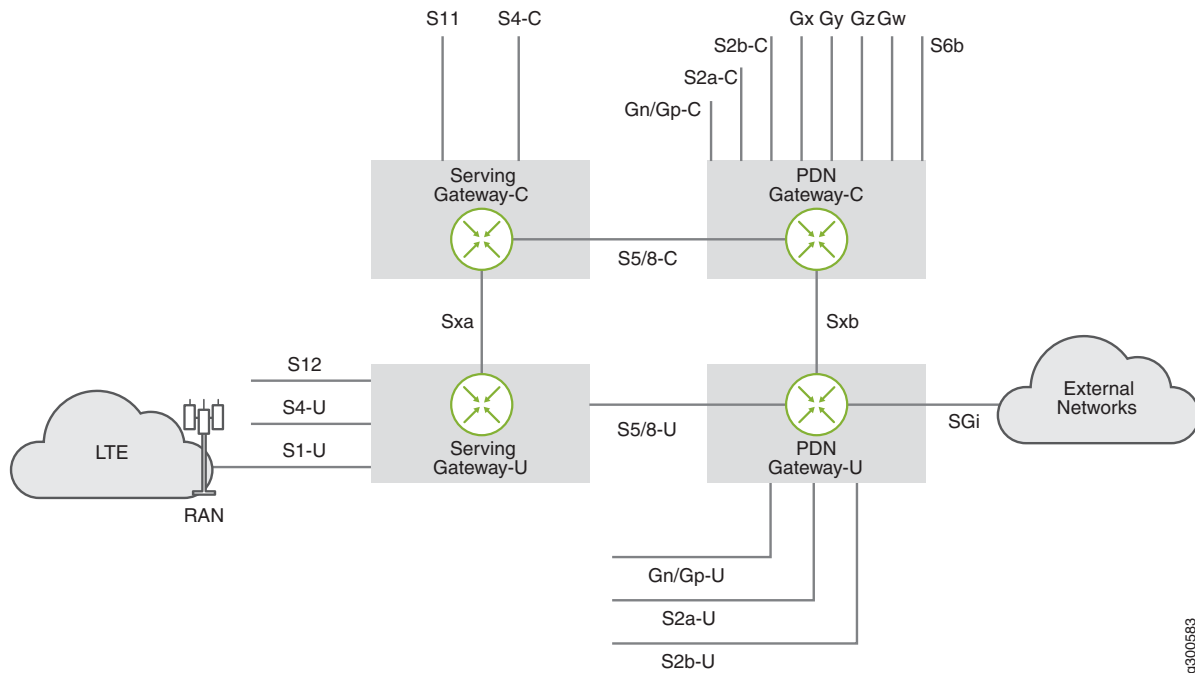
Figure 7: 3GPP Release 8 Evolved Packet Core Architecture



User Equipment (UE) has control and data path connectivity to the EPC network elements over eNodeB base stations. The EPC provides data connectivity to external networks such as the Internet.

3GPP Release 14 introduced CUPS. CUPS stands for Control and User Plane Separation, providing the architecture enhancements for the separation of functionality in the EPC's serving gateway (SGW) and PDN gateway (PGW). As [Figure 8 on page 27](#) shows, both the serving gateway and the PDN gateway of the EPC can be separated into their control plane and user plane functions. CUPS introduces new interfaces, Sxa and Sxb, between the control plane and user plane functions of the SGW and PGW, respectively. CUPS enables control plane and user plane functions to be deployed, scaled and operated separately while integrated over a standard reference interface.

Figure 8: 3GPP Release 14 CUPS Architecture



The control plane provides the following functionality:

- Receives traffic rules and actions
- Triggers accounting (volume and time of traffic)
- Makes session level announcements
- Receives usage information
- Receives user plane status information
- Northbound integration with the signaling plane

The user plane provides the following functionality:

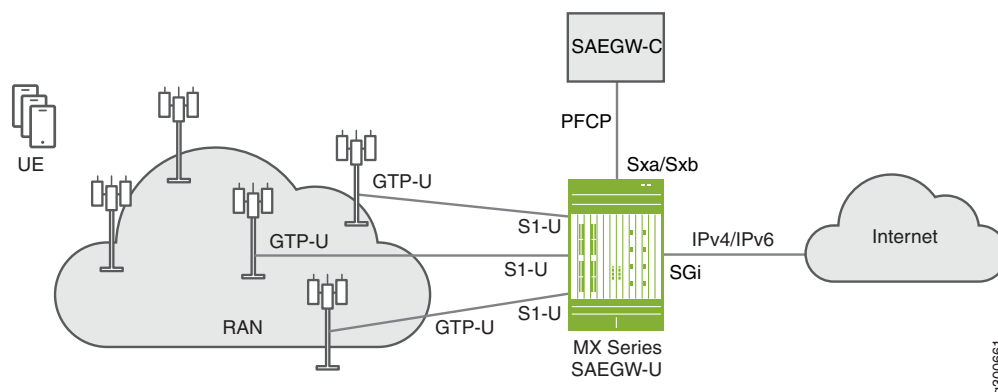
- Subscriber tunnel encapsulations (GTP-U)
- Packet routing and forwarding
- QoS and buffering
- Policy enforcement
- Statistics gathering and reporting
- Optional advanced services

The Junos Multi-Access User Plane solution is to provide a combined SGW user plane (SGW-U) and PGW user plane (PGW-U) in a single MX series router (see [Figure 9 on page 28](#)). The combined SGW-U/PGW-U

is referred to as a SAEGW-U (System Architecture Evolution Gateway-User Plane). Juniper's MX SAEGW-U can interoperate with a third-party combined SGW-C/PGW-C, hereafter referred to as SAEGW-C, through a combined Sxa/Sxb interface.

NOTE: Juniper's MX SAEGW-U communicates with the third-party SAEGW-C over the Sxa/Sxb interface through Packet Forwarding Control Protocol (PFCP) as specified in 3GPP TS 29.244.

Figure 9: Junos Multi Access User Plane SAEGW-U



Benefits of Junos Multi-Access User Plane

With this functional separation, the control plane and user plane have very distinct deployment requirements and can be in different physical locations. While the control plane function is very complex, the user plane function simply requires high packet processing capability and rich policy enforcement. The user plane can be more distributed than the control plane and located closer to end-user access points, such as a radio access network (RAN), enabling higher bandwidth per user while delivering lower latency. Control plane and user plane separation provides the following benefits:

- Independent scaling of the user and control planes.
- Network architecture flexibility including:
 - Ability to deploy from the edge to the core.
 - Ability to segregate different traffic types and services across different user planes while maintaining a common or single control plane.
- Operational flexibility
- Easier migration path from 4G to 5G services. CUPS is optional for 4G, but is an integral part of the 5G network architecture.

WHAT'S NEXT

For more information and full instructions on how to configure Junos Multi-Access User Plane, see [Junos Multi-Access User Plane User Guide](#).

4

CHAPTER

Protect and Reroute Traffic Using BGP Labeled Unicast

How to Protect and Reroute Traffic Using BGP Labeled Unicast | 33

How to Protect and Reroute Traffic Using BGP Labeled Unicast

SUMMARY

Learn how to protect and reroute traffic to the destination network if a border node fails in a multi-domain network.

IN THIS SECTION

- [BGP PIC Edge Using BGP Labeled Unicast Overview | 33](#)

BGP PIC Edge Using BGP Labeled Unicast Overview

SUMMARY

This section talks about the benefits and overview of BGP PIC Edge using BGP labeled unicast as the transport protocol.

IN THIS SECTION

- [Benefits of BGP PIC Edge Using BGP Labeled Unicast | 33](#)
- [How does BGP Prefix Independent Convergence Work? | 34](#)
- [BGP PIC Edge Using BGP Labeled Unicast as the Transport Protocol Overview | 34](#)

Benefits of BGP PIC Edge Using BGP Labeled Unicast

- Provides traffic protection in case of border (ABR and ASBR) node failures in multi-domain networks.
- Provides faster restoration of network connectivity and reduces traffic loss if the primary path becomes unavailable.

How does BGP Prefix Independent Convergence Work?

BGP Prefix Independent Convergence (PIC) improves BGP convergence on network node failures. BGP PIC creates and stores primary and backup paths for the indirect next hop on the Routing Engine and also provides the indirect next hop route information to the Packet Forwarding Engine. When a network node failure occurs, the Routing Engine signals the Packet Forwarding Engine that an indirect next hop has failed, and that the traffic is rerouted to a pre-calculated equal-cost or backup path without modifying BGP prefixes. Routing the traffic to the destination prefix continues by using the backup path to reduce traffic loss until the global convergence through BGP is resolved.

BGP convergence is applicable to both core and edge network node failures. In the case of BGP PIC Core, adjustments to the forwarding chains are made as a result of node or core link failures. In the case of BGP PIC Edge, adjustments to the forwarding chains are made as a result of edge node or edge link failures.

BGP PIC Edge Using BGP Labeled Unicast as the Transport Protocol Overview

BGP PIC Edge using the BGP labeled unicast transport protocol helps to protect and reroute traffic when border nodes (ABR and ASBR) failures happen in multi-domain networks. Multi-domain networks are typically used in Metro Ethernet aggregation and mobile backhaul network designs.

On Juniper Networks MX Series and PTX Series routers, BGP PIC Edge supports IPv4, IPv6, BGP labeled unicast, and Layer 3 VPN services. These BGP services are multipath (learnt from multiple PEs) and resolved through BGP labeled unicast routes, which could again be a multipath learnt from other ABRs. Transport protocols supported over BGP PIC Edge are RSVP, LDP, OSPF, and ISIS.

The following BGP service routes are supported:

- IPv6 Layer 3 VPN services over IPv4 BGP labeled unicast
- IPv4 services over IPv4 BGP labeled unicast
- IPv4 Layer 3 VPN services over IPv4 BGP labeled unicast
- IPv6 BGP labeled unicast service over IPv4 BGP labeled unicast

Starting with Junos OS Release 19.4R1, MX Series and PTX Series routers support BGP PIC Edge with BGP labeled unicast as the transport protocol.

WHAT'S NEXT

For an example on configuring BGP PIC Edge using BGP labeled unicast, see [Example: Protecting IPv4 Traffic over Layer 3 VPN Running BGP Labeled Unicast](#).

For more information on configuring BGP PIC and load balancing, see [Load Balancing for a BGP Session](#).

5

CHAPTER

Defining Flexible Algorithms for Segment Routing in IS-IS

How to Configure Flexible Algorithm in IS-IS for Segment Routing Traffic
Engineering | 37

How to Configure Flexible Algorithm in IS-IS for Segment Routing Traffic Engineering

SUMMARY

A flexible algorithm allows IGP's alone to compute constraint based paths over the network thereby providing simple traffic engineering without using a network controller. This is a light weight solution for networks that have not implemented a controller with full fledged segment routing but still want to reap the benefits of segment routing in their network.

IN THIS SECTION

- [Understanding IS-IS Flexible Algorithm for Segment Routing | 37](#)
- [Configuring Flexible Algorithm for Segment Routing Traffic Engineering | 45](#)

Understanding IS-IS Flexible Algorithm for Segment Routing

SUMMARY

A flexible algorithm allows IGP's alone to compute constraint based paths over the network thereby providing simple traffic engineering without using a network controller. This is a light weight solution for networks that have not implemented a controller with full fledged segment routing but still want to reap the benefits of segment routing in their network.

IN THIS SECTION

- [Benefits of Configuring Flexible Algorithm | 38](#)
- [What is Flexible Algorithm Definition \(FAD\)? | 38](#)
- [Participation in a Flexible Algorithm | 39](#)
- [Network Topology Configured with Flexible Algorithm Definitions | 40](#)
- [Flexible Algorithm RIBs | 44](#)
- [BGP Community and Flexible Algorithms | 44](#)
- [Supported and Unsupported Features | 44](#)

Benefits of Configuring Flexible Algorithm

- A lightweight version of segment routing traffic engineering that can be used in the core of the network.
- Allows you to configure traffic traffic engineering using segment routing even without installing a network controller.
- Utilize equal-cost multipath (ECMP) and TI-LFA per-slice without configuring BGP-LS or static path.
- Compute TI-LFA backup path using the same flexible algorithm definition and constraints computation.
- Take advantage of segment routing traffic engineering using only IS-IS without configuring RSVP or LDP.
- Ability to provision constrained primary path based on a single label.

IGP protocols use a link metric to calculate a best path. However, the best IGP path might not always be the best path for certain types of traffic. Therefore, the IGP computed best path based on the shortest IGP metric is often replaced with traffic engineered path due to the traffic requirements that are not reflected by the IGP metric. Typically RSVP-TE or SR TE is used for computing the path based on additional metrics and constraints to overcome this limitation. Junos installs such paths in the forwarding tables in addition to or as a replacement for the original path computed by the IGP.

Starting in Junos OS Release 19.4R1, you can thin slice a network by defining flexible algorithms that compute paths using different parameters and link constraints based on your requirements. For example, you can define a flexible algorithm that computes a path to minimize IGP metric and define another flexible algorithm to compute a path based on traffic engineering metric to divide the network into separate planes. This feature allows networks without a controller to configure traffic engineering using segment routing without actually implementing a network controller. You can use the prefix SIDs to steer packets along the constraint-based paths. You can configure the prefix SIDs for flexible algorithm through policy configurations.

What is Flexible Algorithm Definition (FAD)?

A flexible algorithm allows IGP to calculate additional best paths based on specified constraints thereby providing simple traffic engineering without using a network controller. This is a lightweight solution for networks that have not implemented a controller with full fledged segment routing but still want to reap the benefits of segment routing in their network. Every operator can define separate constraints or colors depending on their requirements.

To define a flexible algorithm, include **flex-algorithm id** statement at the **[edit routing-options]** hierarchy level. The flexible algorithm definition (FAD) is assigned with an identifier ranging from 128 through 255. This flexible algorithm can be defined on one or more routers in a network. A flexible algorithm computes a best path based on the following parameters:

- **Calculation type**—SPF or strict SPF are the two available calculation type options. You can specify one of these calculation types in your FAD. Select the SPF calculation type if you want to influence the SPF computation on your device based on a certain local policy such as traffic engineering shortcuts. If you select strict SPF then the local policy cannot influence the SPF path selection.
- **Metric type**- IGP metric or TE metric are the available metric type options. You can specify one of these metric types in your FAD depending on your network requirement. If you do not want to use the IGP metric for a specific link you can configure a TE metric that IS-IS can use for calculating the route.
- **Priority**- You can assign a priority to your FADs as per your requirement and IS-IS prioritizes a particular FAD advertisement over another FAD based on your assigned priority.

NOTE: For FADs with link-constraints to work, all relevant links should advertise the admin-colors in IS-IS, which means either RSVP is enabled on the interfaces or **set protocols isis traffic-engineering advertise always** is configured.

- **Set of Link constraints**- You can configure admin-groups for many protocols at the [edit protocols mpls admin-groups] hierarchy level to color an individual link. These **admin-groups** can then be defined as **include any**, **include-all** or **exclude** at the [edit routing-options flex-algorithm definition admin-groups] hierarchy level.

We recommend configuring flexible algorithm on only a few routers to provide redundancy and to avoid conflicts. Flexible algorithm definition is advertised in IGP as **FAD sub-TLVs**. In very large networks, we do not recommend configuring more than 8 flexible algorithm definitions as each flexible algorithm will compute its own path and might cause performance issues beyond that.

The default FAD has the following parameters:

- calculation type: spf
- metric type: igp-metric
- priority: 0
- Link constraints: none

NOTE: Modifying the flexible algorithm definition in a live network or on the fly could cause traffic disruptions until all the nodes converge on the new paths.

Participation in a Flexible Algorithm

You can configure specific routers to participate in a particular flexible algorithm as per your requirement. Paths computed based on a flexible algorithm definition is used by various applications each potentially using its own specific data plane for forwarding the data over such paths. The participating device must

explicitly advertise its participation in a particular flexible algorithm to every application in the segment routing flexible algorithm sub TLV for IS-IS. You can configure a node to participate in a certain flexible algorithm provided it can support the constraints specified in that FAD.

To configure participation in a flexible algorithm include the **flex-algorithm** statement at the **[edit protocols isis source-packet- routing]** hierarchy level. The same device can advertise a FAD and also participate in a flexible algorithm.

Network Topology Configured with Flexible Algorithm Definitions

[Figure 10 on page 41](#) shows the sample topology, there are 8 routers R0, R1, R2, R3, R4, R5, R6, and R7. Four flexible algorithms, 128, 129, 130, and 135 are defined and configured with admin-groups as listed in the following table:

Flex Algorithm Definition (FAD)	Color
128	Include any Red
129	Include any Green
130	Include any Green and Blue
135	Exclude Red

Figure 10: Flexible Algorithm Topology

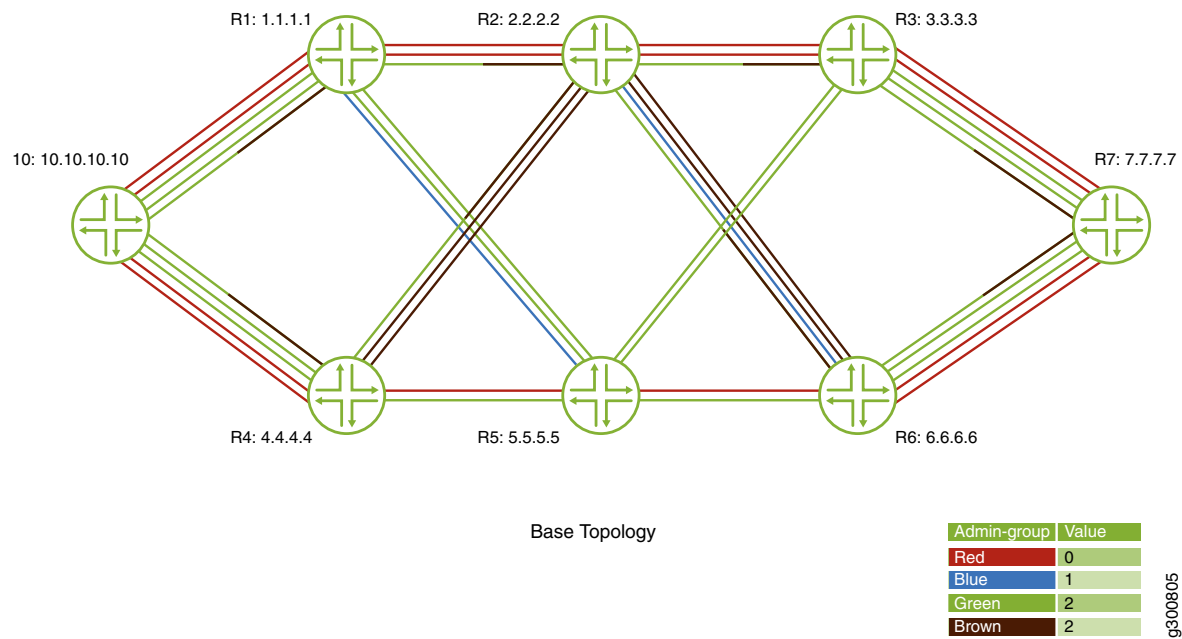


Figure 11 on page 41 shows how FAD 128 routes traffic on any interface that is configured with admin group red.

Figure 11: Traffic Flow for Flexible Algorithm Definition 128

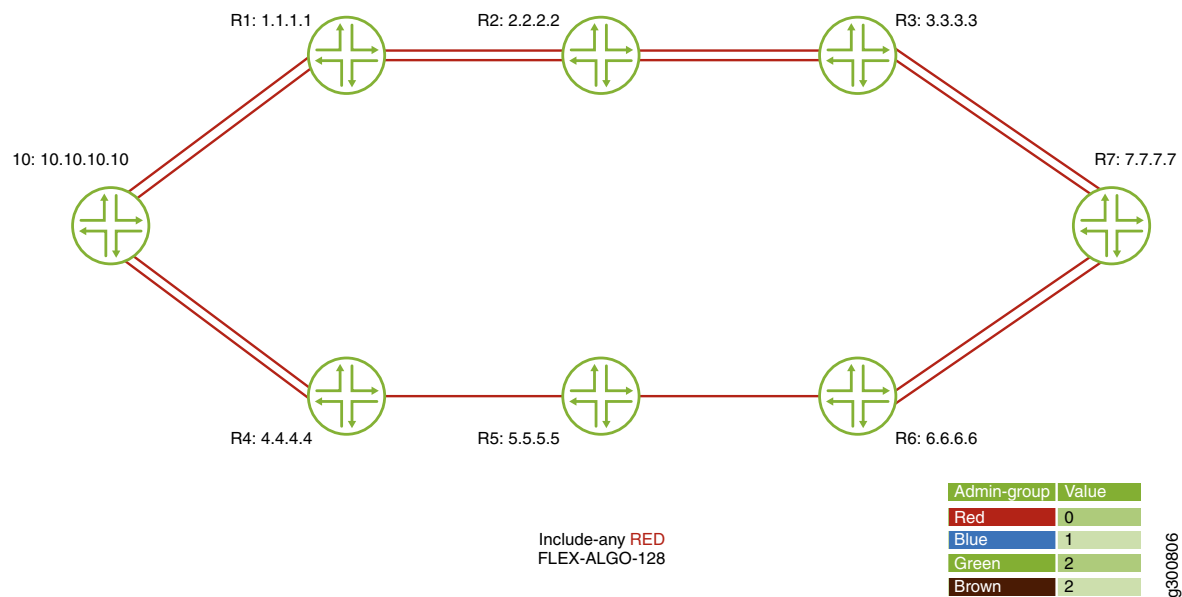


Figure 12 on page 42 shows how FAD 129 routes traffic on any interface that is configured with admin group green.

Figure 12: Traffic Flow for Flexible Algorithm Definition 129

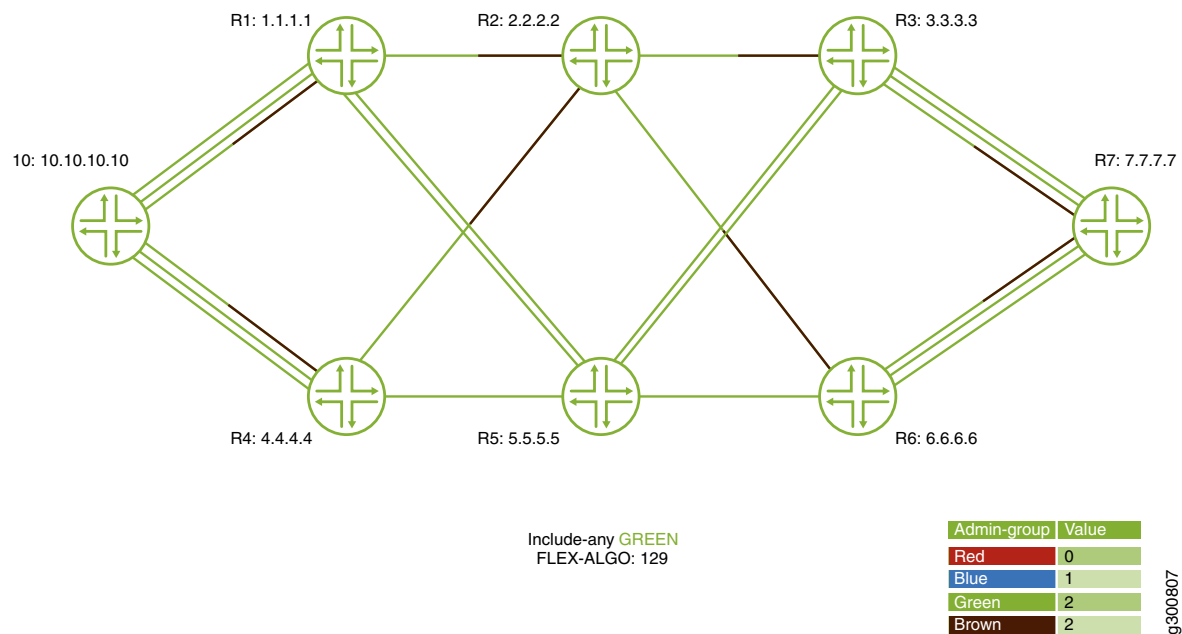


Figure 13 on page 43 shows how FAD 130 routes traffic on any interface that is configured with admin group green and blue.

Figure 13: Traffic flow for Flexible Algorithm Definition 130

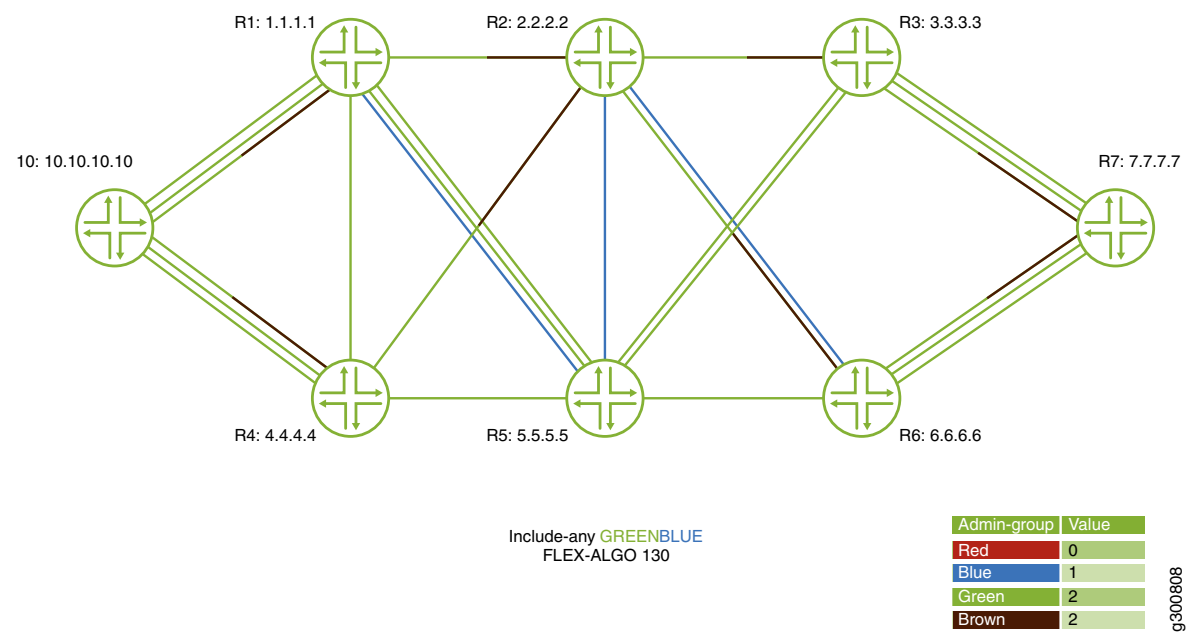
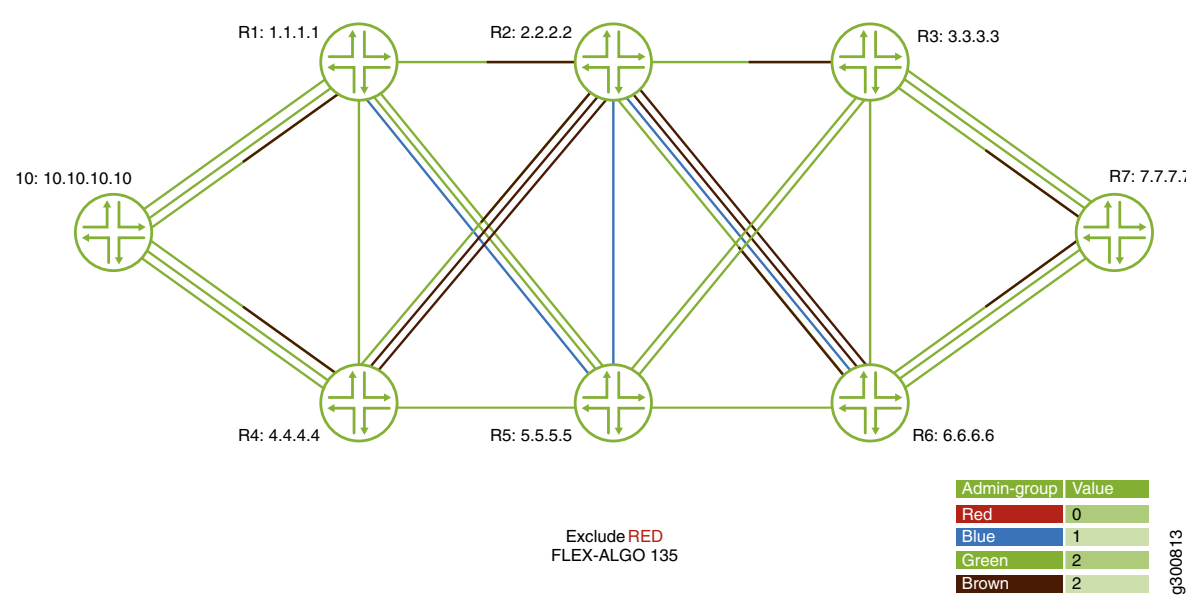


Figure 14 on page 43 shows how FAD 135 routes traffic on any interface that is not configured with admin group red.

Figure 14: Traffic Flow for Flexible Algorithm Definition 135



Flexible Algorithm RIBs

For every flexible algorithm that a router participates in the corresponding flexible algorithm routes are installed in the corresponding flexible algorithm RIB groups also known as routing tables. By default, labeled IS-IS flexible algorithm routes are installed in the `inet.color`, `inet(6)color.0` and `mpls.0` RIBs.

BGP Community and Flexible Algorithms

A flexible algorithm can have an associated BGP color community to resolve routes of other services such as VPN service. By default, the associated BGP color community is the same as the flexible algorithm ID. The flexible algorithm ingress routes that are installed in the `inet(6)color.0` tables will have this color community in the route. However, you can configure a different BGP color community at the `[edit routing-options flex-algorithm id color desired color community value]` hierarchy level.

NOTE: Changing the BGP color community for a flexible algorithm might result in traffic disruption. If you modify a BGP color community for a flexible algorithm then all routes pertaining to that flexible algorithm are removed from the RIB and added again with new colors.

Supported and Unsupported Features

Junos OS supports flexible algorithms in the following scenarios:

- Support for configuring and advertising prefix SIDs for different flexible algorithms.
- Partially supports Internet Draft `draft-ietf-lsr-flex-algo-05.txt` *IGP Flexible Algorithm*

Junos OS does not support the following features in conjunction with flexible algorithms:

- Link delay metric is not supported
- Flexible algorithm is applicable only for default unicast topology, IS-IS multi-topology is not supported.
- IS-IS shortcuts and other IS-IS traffic engineering configuration options are not applicable for flexible algorithm computation
- Inter-level (IS_IS) leaking of flexible algorithm prefix SIDs is not supported.
- Prefix and SID conflict resolution is not supported.
- Remote loop free alternate functionality is not supported because TI-LFA is the preferred FRR computation
- Extended Admin-Groups (EAG) are not supported because they are not supported in IS-IS.

SEE ALSO

*flex-algorithm**definition*[Configuring Flexible Algorithm for Segment Routing Traffic Engineering | 45](#)*show isis flex-algorithm*

Configuring Flexible Algorithm for Segment Routing Traffic Engineering

Before you begin configuring the flexible algorithm for IS-IS, make sure you:

1. Configure the device interfaces to enable IP transport.
2. Configure IS-IS protocol to enable dynamic routing protocol to exchange routing information.
3. Configure BGP protocol.
4. Configure segment routing.

To configure flexible algorithm for IS-IS:

1. Define flexible algorithm on routers that you have identified in your network. Assign a name for the flexible algorithm definition (FAD) ranging from 128 through 255.

```
[edit routing-options ]
user@host# set flex-algorithm name
```

NOTE: We recommend configuring flexible algorithm on only a few routers to provide redundancy and to avoid conflicts.

Specify the parameters of the definition. IS-IS calculates the path based on these specified parameters of the FAD.

- a. Map a BGP color community to the defined FAD. By default each flexible algorithm is associated with a value equal to the flex algorithm.

VPN can be made to resolve paths over the configured BGP color community.

```
[edit routing-options flex-algorithm name]
```

```
user@host# set color desired color community value
```

NOTE: Changing the BGP color community for a flexible algorithm might result in traffic disruption. If you modify a BGP color community for a flexible algorithm then all routes pertaining to that flexible algorithm are removed from the RIB and added again with new colors.

- b. Specify the calculation type based on which the IS-IS protocol calculates the path.

```
[edit routing-options flex-algorithm definition]
user@host# set (spf | strict-spf)
```

- c. Specify the metric type based on which IS-IS calculates the path.

```
[edit routing-options flex-algorithm definition]
user@host# set metric-type (igp-metric | te-metric)
```

- d. Assign a priority level to the advertisement of the FAD based on your requirement. Specify a priority ranging from 0 through 255.

```
[edit routing-options flex-algorithm definition]
user@host# set priority priority
```

NOTE: Modifying the flexible algorithm definition could cause traffic disruptions until all the nodes converge on the new paths.

- e. If you have enabled RSVP traffic engineering, you can configure admin-groups for many protocols to color an individual link.

```
[edit protocols mpls]
user@host# set admin-groups
```

- f. Define the admin groups as per your requirement.

```
[edit routing-options flex-algorithm definition admin-group]
```

```
user@host# set include any admin-group
```

```
user@host# set include-all admin-group
```

```
user@host# set exclude admin-group
```

NOTE: For FADs with link-constraints to work, all relevant links should advertise the admin-colors in IS-IS. You must either enable RSVP on the interfaces or if you have not configured RSVP for traffic engineering, make sure you configure **set traffic-engineering advertise always** at the **[edit protocols isis]** hierarchy level.

2. Identify the participating routers and configure participation on those routers. The same device can advertise a FAD and also participate in a flexible algorithm.

```
[edit protocols isis segment routing]
```

```
user@host# set flex-algorithm
```

3. Advertise prefix segments through policy configuration.

```
[edit policy-options policy-statement name term then]
```

```
user@host# set prefix-segment index index algorithm flex-algo-id node-segment
```

4. To verify if your flexible algorithm configuration is working correctly use the **show isis flex-algorithm** command.

SEE ALSO

[flex-algorithm](#)

[definition](#)

[Understanding IS-IS Flexible Algorithm for Segment Routing | 37](#)

[show isis flex-algorithm](#)

WHAT'S NEXT

For more information on configuring flexible algorithms, see [Example: Configuring IS-IS Flexible Algorithm for Segment Routing](#).

6

CHAPTER

Routing in Fat Tree (RIFT)

How to Integrate RIFT protocol into Junos OS | **51**

How to Integrate RIFT protocol into Junos OS

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 | 51](#)
- [Enabling the RIFT Protocol | 53](#)

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

IN THIS SECTION

- [Benefits of RIFT Protocol | 51](#)
- [RIFT Protocol Overview | 52](#)
- [Impact of Junos Implementation of RIFT Protocol on Network Performance | 53](#)
- [Unsupported Features with RIFT Protocol | 53](#)

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 4 on page 59](#) list the commands you can use to verify the RIFT protocol configuration and status.

Table 4: Commands to Verify RIFT Protocol Configuration

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

WHAT'S NEXT

For more information on configuring the RIFT protocol, see the [RIFT Feature Guide](#).

7

CHAPTER

Wi-Fi Mini-PIM

Wi-Fi Mini-PIM Overview | 65

Wi-Fi Mini-PIM Overview

SUMMARY

The Wi-Fi Mini-Physical Interface Module (Mini-PIM) for SRX Series devices provides an integrated wireless access point (or wireless LAN) solution along with routing, switching, and security in a single device.

IN THIS SECTION

- [Wi-Fi Mini-Physical Interface Module Overview | 65](#)

Wi-Fi Mini-Physical Interface Module Overview

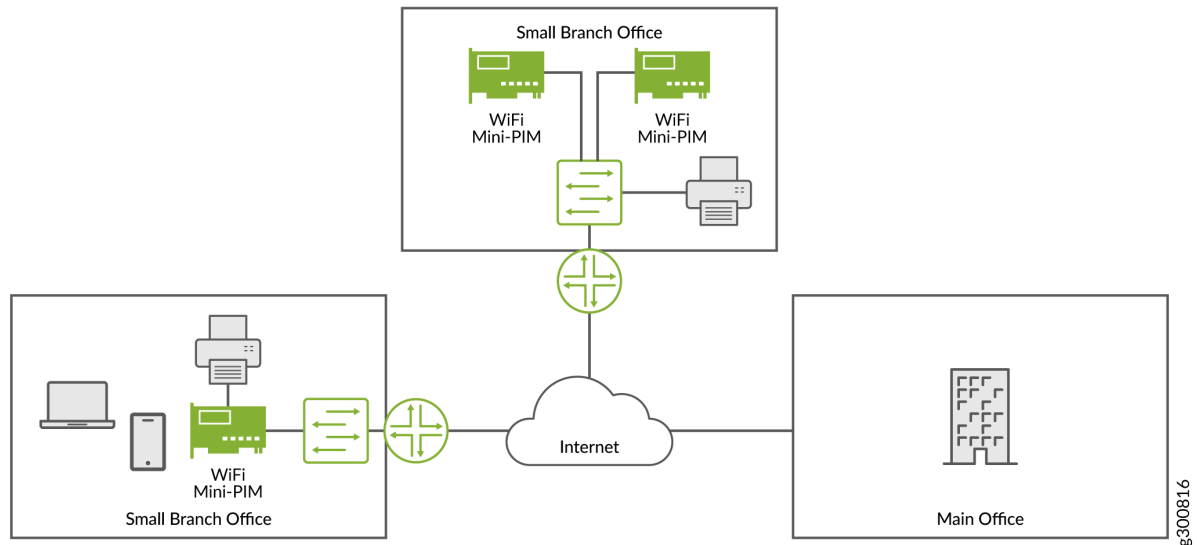
Wi-Fi Mini-Physical Interface Module (Wi-Fi Mini-PIM) for SRX320, SRX340, SRX345, and SRX550M provides an integrated wireless access point—or wireless LAN—along with routing, switching, and security in a single device. Mini-PIM supports the 802.11ac Wave 2 wireless standards and is backward compatible with 802.11a/b/g/n. The Wi-Fi Mini-PIM can coexist with other Mini-PIMs supported on the SRX Series device. [Table 5 on page 66](#) provides a summary of the features supported on Mini-PIM.

Typical deployments for Wi-Fi Mini-PIM solution include:

- Secure wireless LAN connectivity to endpoint devices of corporate users at remote branch offices as seen in [Figure 15 on page 66](#). 802.11ac, WPA2, 802.1X, and SSID-to-VLAN mapping features provide secure Wireless LAN connectivity.
- Direct network connectivity to the enterprise Internet of Things (IoT) devices. The security features on the SRX Series devices secure the IoT devices.

The Mini-PIM inserted onto the SRX series device provides a secure wireless access point connection to multiple clients in a remote location. The following figure illustrates the secure LAN connectivity of Wi-Fi Mini-PIM.

Figure 15: Secure Wireless LAN connection with Wi-Fi Mini-PIM



See [How to Install the Wi-Fi Mini-PIM for SRX Series Services Gateways](#) for more information about how to install the Wi-Fi Mini-PIM.

Features Supported on the Wi-Fi Mini-PIM

[Table 5 on page 66](#) lists the key features supported on the Wi-Fi Mini-PIM.

Table 5: Wi-Fi Mini-PIM Features

Feature	Description
2x2 MU-MIMO	Enables transmission of data to multiple clients simultaneously.
Dual radios	Both radios of 2.4 GHz and 5 GHz bands are simultaneously supported. The maximum supported speed is upto 1.2 Gbps.
Virtual access points (VAPs) and VLAN features	<ul style="list-style-type: none"> Allows you to segment the WLAN into multiple broadcast domains that are the wireless equivalents of Ethernet VLANs. A single access point is segregated into multiple individual VAPs, simulating multiple access points in a single system. An access point supports multiple VLANs, which can be distributed across VAPs and radios. You can configure up to eight VAPs per radio. You can map up to 16 extended service set identifiers (ESSIDs) to individual VLANs. The VLANs from the Mini-PIM software map to VLANs on Junos OS.

Table 5: Wi-Fi Mini-PIM Features (*continued*)

Feature	Description
Co-existence of interfaces	The Wi-Fi Mini-PIM coexists with 4G LTE, VDSL, T1, and serial interfaces.
Client authentication methods	Client authentication methods supported are Wi-Fi Protected Access (WPA) Enterprise (WPA2 standards) and Wi-Fi Protected Access (WPA) Personal (AES-CCMP cipher suits and WPA2 standards).

Benefits of Using the Wi-Fi Mini-PIM

- Provides advanced security capabilities (WPA personal and WPA enterprise authentication methods).
- Simplifies network design, and provides unified management.
- Supports end-to-end segmentation from user to applications with ESSID-to-VLAN mapping.

WHAT'S NEXT

For more information on configuring Wi-Fi Mini-PIM, see [How to Install the Wi-Fi Mini-PIM for SRX Series Services Gateways](#).