Extreme 9920 Software Configuration Guide

21.2.1.0

Table of Contents

Preface

Read the following topics to learn about:

- The meanings of text formats used in this document.
- Where you can find additional information and help.
- How to reach us with questions and comments.

Text Conventions

Unless otherwise noted, information in this document applies to all supported environments for the products in question. Exceptions, like command keywords associated with a specific software version, are identified in the text.

When a feature, function, or operation pertains to a specific hardware product, the product name is used. When features, functions, and operations are the same across an entire product family, such as Extreme Networks switches or SLX routers, the product is referred to as *the switch* or *the router*.

Table 1: Notes and warnings

Icon	Notice type	Alerts you to...
💡	Tip	Helpful tips and notices for using the product
📝	Note	Useful information or instructions
➡️	Important	Important features or instructions
⚠️	Caution	Risk of personal injury, system damage, or loss of data
⚠️	Warning	Risk of severe personal injury

Table 2: Text

Convention	Description
`screen displays`	This typeface indicates command syntax, or represents information as it is displayed on the screen.
The words *enter* and *type*	When you see the word *enter* in this guide, you must type something, and then press the Return or Enter key. Do not press the Return or Enter key when an instruction simply says *type*.
Key names	Key names are written in boldface, for example **Ctrl** or **Esc**. If you must press two or more keys simultaneously, the key names are linked with a plus sign (+). Example: Press **Ctrl**+**Alt**+**Del**
Words in italicized type	Italics emphasize a point or denote new terms at the place where they are defined in the text. Italics are also used when referring to publication titles.
NEW!	New information. In a PDF, this is searchable text.

Table 3: Command syntax

Convention	Description
bold text	Bold text indicates command names, keywords, and command options.
italic text	Italic text indicates variable content.
`[ ]`	Syntax components displayed within square brackets are optional. Default responses to system prompts are enclosed in square brackets.
`{ x	y	z }`	A choice of required parameters is enclosed in curly brackets separated by vertical bars. You must select one of the options.
`x	y`	A vertical bar separates mutually exclusive elements.
`< >`	Nonprinting characters, such as passwords, are enclosed in angle brackets.
…	Repeat the previous element, for example, `member[member...]`.
`\`	In command examples, the backslash indicates a "soft" line break. When a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

Documentation and Training

Find Extreme Networks product information at the following locations:

Current Product Documentation

Release Notes

Hardware and Software Compatibility for Extreme Networks products

Extreme Optics Compatibility

Other Resources such as articles, white papers, and case studies

Open Source Declarations

Some software files have been licensed under certain open source licenses. Information is available on the Open Source Declaration page.

Training

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For details, visit the Extreme Networks Training page.

Help and Support

If you require assistance, contact Extreme Networks using one of the following methods:

Extreme Portal

Search the GTAC (Global Technical Assistance Center) knowledge base; manage support cases and service contracts; download software; and obtain product licensing, training, and certifications.

The Hub

A forum for Extreme Networks customers to connect with one another, answer questions, and share ideas and feedback. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.

Call GTAC

For immediate support: (800) 998 2408 (toll-free in U.S. and Canada) or 1 (408) 579 2800. For the support phone number in your country, visit www.extremenetworks.com/support/contact.

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number, or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any actions already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)
- Network load at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

Subscribe to Product Announcements

You can subscribe to email notifications for product and software release announcements, Field Notices, and Vulnerability Notices.

1. Go to The Hub.
2. In the list of categories, expand the **Product Announcements** list.
3. Select a product for which you would like to receive notifications.
4. Select **Subscribe**.
5. To select additional products, return to the **Product Announcements** list and repeat steps 3 and 4.

You can modify your product selections or unsubscribe at any time.

Send Feedback

The User Enablement team at Extreme Networks has made every effort to ensure that this document is accurate, complete, and easy to use. We strive to improve our documentation to help you in your work, so we want to hear from you. We welcome all feedback, but we especially want to know about:

- Content errors, or confusing or conflicting information.
- Improvements that would help you find relevant information.
- Broken links or usability issues.

To send feedback, email us at documentation@extremenetworks.com.

Provide as much detail as possible including the publication title, topic heading, and page number (if applicable), along with your comments and suggestions for improvement.

What's New in this Document

The following table describes the changes to this guide for the Extreme 9920 software, release 21.2.1.0.

Table 4: Summary of changes

Feature	Description	Link
400G Line Card Support	Extreme 9920 software supports 400G speed on the line card.	• Limitations on page 12 • Line Card on page 81 • Configure Line Card on page 81

For more information about this release, see the *Extreme 9920 Software Release Notes, 21.2.1.0*.

Introduction to the NPB Application

The NPB software runs the NPB operating system and provides network packet broker functions.

NPB Application Overview

The NPB application provides network packet broker functions to process and prepare packets for visibility tools. This allows core networking devices to offload network monitoring.

When the Extreme 9920 device running Extreme 9920 software is attached to optical taps between the core networking devices, a copy of the traffic is sent to the Extreme 9920 device for filtering the traffic of interest, formatting it, and sending it on to visibility tools.

The NPB application supports the following network packet broker functions:

- Link Aggregation: Aggregates traffic arriving from multiple ports and directs it to a single port or port-channel ("many to one").

  Link Aggregation Control Protocol (LACP): Enables devices to send Link Aggregation Control Protocol Data Units (LACPDUs) to each other and establish link aggregation connections.

- Replication: Replicates network traffic to multiple ports and port-channels ("one to many").

- Load balancing: Distributes network traffic among ports in a port-channel.

- ACL filtering: Directs network traffic based on Layer 2 to Layer 4 protocol headers.

- Route-map forwarding: Redirects packets based on Layer 2 to Layer 4 Protocol headers to the desired physical port or port-channel interfaces.

- Packet slicing: Truncates the packet to a specified length.

- Tunnel origination or encapsulation: Encapsulates packets with IPv4 Generic Routing Encapsulation (GRE) headers.

- Tunnel termination: Classifies and decapsulates incoming IPv4 and IPv6 packets.

- Encapsulation-header stripping:
  ◦ Removes tags that are not supported by visibility applications.

- Supports 802.1BR, VN-Tag, VLAN, VXLAN, GTPU, GRE, and IPIP headers.
- Quality of Service (QoS): Provides preferential treatment to specific traffic that is received on multiple ingress interfaces or Test Access Points (TAPs).
- Link Layer Discovery Protocol (LLDP): Allows network devices to advertise their identity and capabilities to peer devices on a LAN and store information about the network.

Supported Device Information

The Extreme 9920 software with the NPB application runs on Extreme 9920 devices.

Limitations

Extreme 9920 software 21.2.1.0 has the following limitations:

- Filtering based on 802.1BR tag is not supported, only the inner filter after 802.1BR/VN tag is supported.
- IPv6 Transport tunnel is not supported.
- Only one precedence per egress is supported.
- The first level of parsing of Non-Transport tunnel is supported.
- Route map can be present in only one ingress group having only ports.
- Only 400G speed is supported on the line card.
- Breakout is not supported for 400G ports.

Network Traffic Monitoring

Packet forwarding and filtering are core features of Extreme 9920 software. To properly configure and use the Extreme 9920 effectively, you need a good understanding of the various components that configure the packet workflow to monitor your network.

Before you configure any settings, you must create components that monitor and modify packets flowing into and out of the Extreme 9920.

> **Note**
> For the purpose of configuration, it is recommended that you adhere to the following configuration order when you create these building blocks. When you have a basic packet workflow in place, modifications to that workflow follow the same order as outlined here.

1. Access-control lists (ACLs)
2. Encapsulations for any tunnel origination
3. Listener-policies
4. Egress
5. Egress-groups
6. Route-maps
7. Interfaces
8. Ingress-groups
9. Replication
10. Header modification

11. Packet Truncation and Termination

12. Tunnel Termination

It is important to note the terminology used when discussing packet flow through the 9920. Interfaces that are connected to the network being monitored are termed *tap interfaces*. Interfaces connected toward the visibility tools are called *tool ports*. Because tap ports often receive the same traffic types, you create and configure an *ingress-group* to group the tap interfaces to the same desired *route-map*.

Likewise, tool ports are configured in an *egress*, which combines interfaces and any post-processing directives (such as dropping specific packets, removing tunnel or specific headers, adding a VLAN tag, or truncating packet length) by using a listener-policy. Between the 'ingress-group' and the 'egress' is an abstraction layer that connects the two together, the 'egress-group'.

You can configure various actions at each level, and some could depend on a prior entity existing in the configuration. The examples that are provided are not exhaustive for all possible configurations or parameters. For additional information, see *Extreme 9920 Software Command Reference, 21.2.1.0*.

Access Control List

An access control list (ACL) is a set of rules defined to filter the network traffic. Each ACL is assigned a unique name.

Packet filtering and traffic flow through the network are managed with ACLs, which contain rules that you configure for that purpose. Incoming packets are matched against the entries in ACL. Packets are forwarded or dropped based on criteria specified in ACL. The unique sequence number of each entry indicates the order that the packet will be matched against rules in the ACL. The lower the sequence ID, the earlier the rule will be checked against the packet.

> **Note**
> Care should be taken when designing the ACLs being used to prevent a lower-sequence ID from matching all the traffic desired for a higher sequence ID.

ACLs are classified as MAC (Layer 2), IPV4 (Layer 3), or IPv6 (Layer 3) access lists, based on the matching keys. If incoming packets match both Layer 2 and Layer 3 ACLs, Layer 3 ACLs in the same route-map stanza are prioritized and actions associated with L3 ACLs are applied.

- ACLs that are referenced in route-maps or listener polices can be modified and deleted.
- When ACL entries that are in use are deleted, the associated route-map or listener-policy re-program the hardware accordingly.
- Each ACL entry can specify its ability to count or log:
  ◦ The counting action provides packet and octet count (64-bit capacity).
  ◦ Logging action sends a copy of the frame to the CPU.
  ◦ The forwarding action remains unchanged.

Create an IPv4 Access Control List

You can create an IPv4 Access Control List (ACL), which you can configure with rules that permit or deny traffic based on packet fields belonging to the IPv4 family of protocols.

About This Task
Note the following naming conventions for name identifiers:

- The name cannot exceed 64 characters.
- The name must start with an alphabet character or an underscore.
- The name must contain alphanumeric characters and special characters (underscores, hyphens, or periods).
- The following reserved keywords cannot be used as name identifiers: `all`, `ingress-group`, `egress`, `egress-group`, `match`, `list`, `access-list`, `route-map`, and `listener-policy`.

Procedure

1. Enter the Config mode.

   ```
   device(config)#
   ```

2. Create an IPv4 ACL.

   ```
   device(config)# ip access-list acl-name
   device(config)# ip access-list acl5-ipv4
   device(config-ip-acl)# permit ip any any
   device(config-ip-acl)# end
   device#
   ```

3. Ensure that the ACL is created.

   ```
   device#show ip access-list all
   ip access-list acl5-ipv4
     seq 10 permit ip any any
   ( 0 Packets, 0 Bytes, 0 Packets/sec, 0 Bits/sec )
   ```

Create an IPv6 Access Control List

You can create an IPv6 access list, which you can configure with rules that permit or deny traffic based on packet fields of the IPv6 family of protocols.

About This Task
Note the following naming conventions for name identifiers:

- The name cannot exceed 64 characters.
- The name must start with an alphabet character or an underscore.
- The name must contain alphanumeric characters and special characters (underscores, hyphens, or periods).
- The following reserved keywords cannot be used as name identifiers: `all`, `ingress-group`, `egress`, `egress-group`, `match`, `list`, `access-list`, `route-map`, and `listener-policy`.

Procedure

1. Enter the Config mode.

```
device(config)#
```

2. Create an IPv6 ACL.

```
device(config)# ipv6 access-list acl-name
device(config)# #ipv6 access-list P6
device(config-ipv6-acl)# permit ipv6 any any vlan 500
device(config-ipv6-acl)# end
device#
```

3. Ensure that the access control list is created.

```
device# show ipv6 access-list all
ipv6 access-list P6
   seq 10 permit ipv6 any any vlan 500
( 0 Packets, 0 Bytes, 0 Packets/sec, 0 Bits/sec )
```

Create a MAC Access Control List

You can create a MAC access control list, which you can configure with rules that permit or deny traffic based on packet fields of the L2 OSI layer.

About This Task

Note the following naming conventions for name identifiers:

- The name cannot exceed 64 characters.
- The name must start with an alphabet character or an underscore.
- The name must contain alphanumeric characters and special characters (underscores, hyphens, or periods).
- The following reserved keywords cannot be used as name identifiers: `all`, `ingress-group`, `egress`, `egress-group`, `match`, `list`, `access-list`, `route-map`, and `listener-policy`.

Procedure

1. Enter the Config mode.

```
device(config)#
```

2. Create a MAC ACL.

```
device(config)# mac access-list acl-name
device(config)# mac access-list L2
device(config-mac-acl)# permit any any vlan 350
device(config-mac-acl)# end
device#
```

3. Verify that the access control list is created.

```
device# show mac access-list all
mac access-list L2
  seq 10 permit any any vlan 350
( 0 Packets, 0 Bytes, 0 Packets/sec, 0 Bits/sec )
```

Encapsulation

Generic Routing Encapsulation (GRE) headers provide a private secure path for transporting packets.

You can configure outgoing-packets encapsulation with configured tunnel and L2 or L3 parameters. The following L2 and L3 parameter configurations are required for GRE encapsulation:

- Outer destination MAC address (mandatory)
- Outer source MAC address (mandatory)
- Outer VLAN Tag (optional)
- Outer VLAN PCP (optional)
- Outer Source IPv4 address (mandatory)
- Outer Destination IPv4 address (mandatory)

> **Note**
> The flags in the GRE encapsulation header are set to 0.

Configure Encapsulation

About This Task
The L2 and L3 parameters are configured as part of encap object, and the encap object is attached to an egress object to encapsulate packets.

Procedure

1. Enable encapsulation.
   ```
   # encap encap-name
   ```
2. Configure the encapsulation type.
   ```
   # encap-type [ gre | erspan ]
   ```
3. Configure the source IPv4 or MAC address.
   - Source IPv4 address:
   ```
   # source-ipv4-addr ip-addr
   ```
   - Source MAC address:
     ```
     # source-mac-addr mac-addr
     ```
4. Configure the destination IPv4 or MAC address.
   - Destination IPv4 address:
   ```
   # destination-ipv4-addr ip-addr
   ```
   - Destination MAC address:
     ```
     # destination-mac-addr mac-addr
     ```
5. Configure the VLAN ID.
   ```
   # vlan-id vlan-id-value
   ```
6. Configure the VLAN priority (PCP) value.
   ```
   # vlan-pcp vlan-pcp-value
   ```

Example

```
encap encap1
  source-mac-addr 00:00:00:11:11:11
  destination-mac-addr 00:00:00:22:22:22
  source-ip-addr 1.1.1.1
  destination-ip-addr 2.2.2.2
  vlan-id 100
  vlan-pcp 3

interface ethernet 1/1
   no  shutdown

interface ethernet 1/2
   no shutdown

interface ethernet 1/1
   ingress-group ingg1

egress eg1
set encap encap1
   precedence 10  interface ethernet 1/2
quit

set egress-group egg1
     egress eg1
quit

ip access-list v4
 seq 10 permit ip any any
quit

route-map rm1 10
   match ip access-list v4
   egress-group egg1
quit

ingress-group ingg1
   route-map rm1
quit
```

Listener Policy

A listener policy is a set of commands that specifies the actions to be performed on packets as they leave the 9920 device. Each listener policy can exist as multiple instances, which are differentiated by a user-specified, unique sequence ID. Listener policies are associated with other settings that are also specified as policies.

The NPB application supports multiple listener policies, each uniquely identified by a listener policy name. Each listener policy can have multiple stanzas, which are uniquely identified by the sequence ID. A listener policy applies an ACL to an egress and the interfaces associated with it, and defines the actions for the matching ACL. When a listener policy is bound to the egress, it applies the listener policy actions to packet flow.

Listener policies support the following actions:
- Permit/deny traffic
- Packet truncation
- Tunnel termination

- Tag stripping
- VLAN tag addition

Configure a Listener Policy

Perform this procedure to map an ACL to an ingress policy and an egress and define actions for a matching ACL.

About This Task
Each listener policy can exist as multiple instances, which are differentiated by a user-specified, unique sequence ID. A listener policy maps an ACL of each type to an egress and defines the actions for the matching ACL.

Procedure

1. Enter the Config mode.
   ```
   device(config)#
   ```

2. Configure the access list and actions.
   ```
   device(config)# ip access-list acl-name
   ```

3. Create the listener policy, match the ACL, and include any action subcommands for the policy.
   ```
   device(config)# listener-policy lp-2
   device(config-listener-policy)# match ip access-list acl5-ipv4

   device(config-listener-policy)# strip-brtag
   device(config-listener-policy)# vlan vl-4085
   device(config-listener-policy)# description "ABCD"
   ```

4. Configure an egress policy, and bind the listener policy, specifying any additional egress actions.
   ```
   device(config-egress)# egress e2
   device(config-egress)# set listener-policy lp-2
   device(config-egress)# precedence 1 interface ethernet 1/14
   ```

5. Configure an egress group and associate it with the egress policy.
   ```
   device(config-egress-group)# egress-group eg_1
   device(config-egress-group)# description egress-group_1
   device(config-egress-group)# set egress e2
   ```

6. Configure the route map, and set any other parameters, such as forwarding actions, match ip access list, and the egress-group.
   ```
   device(config-route-map)# route-map R1 10
   device(config-route-map)# forward-action permit
   device(config-route-map)# match ip access-list test_1
   device(config-route-map)# set egress-group eg_1
   ```

7. Configure the interface port and channel for egress traffic.

   > **Note**
   > In the following example, traffic is leaving on slot/port number 2/14.

   ```
   interface ethernet 2/14
   speed 100000
   description To_Tool
   no shutdown
   ```

8. Configure the interface port and channel for ingress traffic.

> **Note**
> In the following example, traffic is coming in on slot/port number 2/3.

```
interface ethernet 2/3
description From_TAP
ingress-group TAP_TRAFFIC
no shutdown
```

9. Configure an ingress group and associate a route map.

```
ingress-group TAP_TRAFFIC
set route-map R1 10
```

Create a Listener Policy

Follow this procedure to create a listener-policy, which is used by an egress to combine interfaces and perform post-processing actions.

About This Task
Note the following naming conventions for name identifiers:

- The name cannot exceed 64 characters.
- The name must start with an alphabet character or an underscore.
- The name must contain alphanumeric characters and special characters (underscores, hyphens, or periods).
- The following reserved keywords cannot be used as name identifiers: `all`, `ingress-group`, `egress`, `egress-group`, `match`, `list`, `access-list`, `route-map`, and `listener-policy`.

> **Note**
> Sequence ID is required when you create a listener policy; it defines the order in which it is to be processed (see the *Extreme 9920 Software Command Reference, 21.2.1.0* for more information).

Procedure

1. Enter the Config mode.
```
device(config)#
```

2. Create a listener policy.
```
device(config)#listener-policy lpl 10
device(config-listener-policy)# match ip access-list v4_jumbo_traffic
device(config-listener-policy)# truncate 512
device(config-listener-policy)# end
device#
```

3. Verify that the listener policy is created.
```
device# show listener-policy lp1
show listener-policy lpl
listener-policy lpl 10
forward-action permit
match ip access-list v4_jumbo_traffic (pending)
truncate 512
Policy matches: 0  packets, 0  bytes, 0 Packets/sec, 0 Bits/sec
```

Egress

An egress defines the interface or range of interfaces to be used for tool ports. You can create an egress and combine it with various policies to perform the required actions on packets at the egress.

Tool ports are entered into an egress, which can use a listener-policy to apply some additional processing actions to packets leaving the interfaces of the egress.

Egress Policy

An egress policy is used to specify actions to be taken at egress on packets exiting 9920 to prepare them for downstream tools. You must configure a match ACL and a listener policy, at a minimum.

Egress policies are associated with ACLs, route-maps, listener policies, and other settings.

> **Note**
> It is important to note the following when configuring an egress policy:

- An egress policy can specify only one listener policy.
- An ACL bound to an egress policy can be modified.
- A listener policy bound to an egress can be modified.
- An ACL bound to a listener policy or route-map can be modified.

Create an Egress

Before You Begin
Note the following naming conventions for name identifiers:
- The name cannot exceed 64 characters.
- The name must start with an alphabet character or an underscore.
- The name must contain alphanumeric characters and special characters (underscores, hyphens, or periods).
- The following reserved keywords cannot be used as name identifiers: `all`, `ingress-group`, `egress`, `egress-group`, `match`, `list`, `access-list`, `route-map`, and `listener-policy`.

About This Task
Perform this procedure to define tool ports for egress packet flow in an egress.

> **Note**
> A port can either be an individual port or a member port-channel.
> A port associated with an egress configuration cannot be configured as a member port-channel until its active configurations are cleared.

Procedure

1. Enter the Config mode.

```
device(config)#
```

2. Create an egress.

```
device(config)# egress egress-one
device(config-egress)# precedence 10 interface ethernet 1/10
device(config-egress)# end
device#
```

3. Verify that the egress is created.

```
device# show egress egress-one
    Name : Egress egress-one
    Precedence 10 interface ethernet 1/10
```

Configure an Egress Policy

About This Task

Follow this procedure to prepare packets for actions to be performed when leaving the 9920.

Procedure

1. Enter the Config mode.

```
device# configure terminal
device(config)#
```

2. Configure an ACL of type IPv4, IPv6, or MAC and any actions.

```
device(config)# ip access-list acl5-ipv4
device(config-ip-acl)# seq 15 permit ip any any count
```

The specified ACL and configured actions are bound to a listener policy.

3. Create the listener policy, including any action subcommands for the policy.

> **Note**
> A listener policy supports only one IPv4 ACL, IPv6 ACL, or MAC ACL.

```
device(config)# listener-policy lp-2 220
device(config-listener-policy)# match ip access-list acl5-ipv4
device(config-listener-policy)# strip vn-tag
device(config-listener-policy)# description "ABCD"
```

4. Configure an egress policy, and bind the listener policy, specifying any additional egress actions.

> **Note**
> An egress can be associated with only one listener policy.

```
device(config-egress)# egress e2
device(config-egress)# set listener-policy lp-2
device(config-egress)# description DirectTool
device(config-egress)# set encap encap-1
device(config-egress)# precedence 1 interface ethernet 2/14
```

5. Configure the interface port and channel for egress traffic.

> **Note**
> In the following example, traffic is leaving on slot/port number 2/14.

```
interface ethernet 2/14
speed 100000
description To_Tool
no shutdown
```

Egress-Group

An egress-group is a mechanism that connects an ingress-group and the egress to define how traffic is forwarded to end devices.

Create an Egress Group

Egress group specifies where the packets from the 9920 tap interfaces are to be copied for the various tools.

Before You Begin
Note the following naming conventions for name identifiers:

- The name cannot exceed 64 characters.
- The name must start with an alphabet character or an underscore.
- The name must contain alphanumeric characters and special characters (underscores, hyphens, or periods).
- The following reserved keywords cannot be used as name identifiers: `all`, `ingress-group`, `egress`, `egress-group`, `match`, `list`, `access-list`, `route-map`, and `listener-policy`.

Procedure

1. Enter the Config mode.
   ```
   device(config)#
   ```

2. Create an egress group.
   ```
   device(config)# egress-group egg123
   device(config-egress-group)# set egress egress-one
   device(config-egress-group)# end
   device#
   ```

3. Verify that the egress-group is created.
   ```
   device# show running-config egress-group
     egress-group egg123
     egress-one
   ```

Route-Map

A route-map consists of ACLs and set of directives that define the egress interfaces for forwarding the flow. A route-map evaluates incoming packets and determines the

traffic of interest for the tools based on the ACLs used. Any additional processing of the packets by the route-map is reflected on the egress groups used by route-map

A route-map consists of a sequence of instances, equivalent of rows in a table. The device evaluates an incoming packet according to route-map instances in ascending numerical order. The incoming packet is first compared against instance 1, then against instance 2, and so on. When the device finds a match, the device stops evaluating the incoming packet.

Route-maps contain `match` clauses and `set` statements. Each route-map contains a `forward-action permit` or `forward-action deny` statement that modifies the behavior of the route-map instance:

- If the route-map contains a `forward-action permit` statement, the access lists present in the match statement match against incoming packets, and forwards the matched packets according to the access lists. If the packets do not match the access lists in the match statement, the packets pass to the next instance in the same. route-map. If the packet does not match any of the access lists in the route-map, it is dropped.
- If the route-map instance contains a `deny` statement, the matching route-map instance is skipped. The route-map instance does not program the access lists in the match statements, and packets that match, skip that route-map instance.
- If an incoming packet does not match any access lists in the match statements of the route-map, the packet is dropped. This is the default action.
- To change the default action, configure the last `match` statement in the last instance of the route-map to a MAC access list with a clause of `permit any any`.
- If there is no `match` statement, route-map instance is skipped like the `forward-action` is set to `deny`.

If the route-map contains `set` statements, packets that are permitted by the route-map `match` statements are forwarded according to the `set` statements.

Create Egress Group

Procedure

1. Create an egress and add (remove) port interface to the egress.

```
device (config)# egress name
device (config)# [no] precedence 10 interface ethernet 0/10
device (config)# [no] set encap encap100
device (config)# [no] set listener-policy lp100
```

2. (Optional) Delete an egress port.

```
device (config)# no egress name
```

3. Group the egress ports associated with the egress group.

```
device (config)# egress-group name
device (config)# set egress egress-one
device (config)# set egress egress-two
```

Example

Adding second interface to an egress is not supported.

```
device (config)# egress eg-test1
device (config)# [no] precedence 10 interface ethernet 1/4
device (config)# [no] precedence 10 interface ethernet 1/4

Error: precedence already exists ethernet 1/4 egress eg-test1.
```

Create a Route-Map

You can create a route-map, that evaluates packets and specifies actions to be performed on them.

About This Task

Note the following naming conventions for name identifiers:

- The name cannot exceed 64 characters.
- The name must start with an alphabet character or an underscore.
- The name must contain alphanumeric characters and special characters (underscores, hyphens, or periods).
- The following reserved keywords cannot be used as name identifiers: `all`, `ingress-group`, `egress`, `egress-group`, `match`, `list`, `access-list`, `route-map`, and `listener-policy`.

> **Note**
> Sequence ID is required when you create a route-map; it defines the order in which it is to be processed (see the *Extreme 9920 Software Command Reference, 21.2.1.0* for more information).

When you create a route map, you associate it with an egress-group using the **set** command.

Procedure

1. Enter the Config mode.

   ```
   device(config)#
   ```

2. Create a route map.

   ```
   device(config)# route-map noc_traffic 10
   device(config-route-map)# match ip access-list v4_external
   device(config-route-map)# set egress-group netscout
   device(config-route-map)# end
   device#
   ```

3. Verify that the egress is created.

   ```
   device# show route-map all
     rmap1 10
     egress-group egg123
   ```

Map ACLs with Route-Map

Procedure

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Create ACL and define the rules.

   ```
   device(config)# ip access-list name
   ```

3. Create a route-map.

   ```
   device (config)# route-map name sequence_number
   ```

4. (Optional) Delete the route-map instance.

   ```
   device (config) # no route-map name
   ```

5. Map ACL to the route-map.

   ```
   device (config-route-map-name) # match [ mac | ip | ipv6 ] access-list name
   ```

6. (Optional) Remove ACL mapping from the route-map.

   ```
   device (config-route-map-name) # no match [ mac | ip | ipv6 ] access-list name
   ```

7. Configure the route-map to be matched or skipped.

   ```
   device (config-route-map-name) # forward-action [ permit | deny ]
   ```

8. Configure the egress-group.

   ```
   device (config-route-map-name) # set egress-group name
   ```

Interfaces

Extreme 9920 interfaces can operate in various modes. After enabling the interfaces, the ingress-group that matches the traffic of interest coming from the monitored network can be applied.

Create an Interface

You can create an interface and associate it with an ingress group to apply various policies to the traffic at the tool ports.

Before You Begin
Interface name must be in slot/port format. For more information, see the *Extreme 9920 Software Command Reference, 21.2.1.0*.

Procedure

1. Enter the Config mode.

   ```
   device(config)#
   ```

2. Create an interface or interface range.

   ```
   device(config)# interface e 1/10-14,3/3-4
   device(config-if-eth-1/10-14,3/3-4)# no shutdown
   device(config-if-eth-1/10-14,3/3-4)# set ingress-group noc_traffic
   device(config-if-eth-1/10-14,3/3-4)# end
   ```

3. Verify that the interface configuration was accepted.

```
device# show ingress-group noc_traffic

   Name : noc_traffic
      Route-Map : -
    Description : -
     Interfaces :  ethernet 1/10  ethernet 1/11
        ethernet 1/12  ethernet 1/13  ethernet 1/14
        ethernet 3/3  ethernet 3/4
```

4. View the interface status.

```
device# show interface ethernet 1/10-14,3/3-4 | include state
ethernet 1/10 Admin state UP     Operational state UP
ethernet 1/11 Admin state UP     Operational state UP
ethernet 1/12 Admin state UP     Operational state UP
ethernet 1/13 Admin state UP     Operational state UP
ethernet 1/14 Admin state UP     Operational state UP
ethernet 3/3 Admin state UP      Operational state UP
ethernet 3/4 Admin state UP      Operational state UP
```

IP Addresses

IP address is an Internet Protocol used to deliver packets of data from a source to a destination across an interconnected system of networks.

An IP address has two sections:

- Network: Identifies the network on which the device is configured.
- Host: Identifies the host device.

IPv4 Address

IPv4 uses a fixed-length 32-bit addressing system and is represented in a 4-byte dotted decimal format, x.x.x.x.

IPv6 Address

IPv6 increases the number of network address bits from 32 (IPv4) to 128 bits, which provides more unique IP addresses to support more network devices.

An IPv6 address consists of 8 fields of 16-bit hexadecimal values separated by colons (:), x:x:x:x:x:x:x:x

Configure an IP Address

Procedure

1. Enter the Config mode.

```
device# configure terminal
```

2. Access the management interface.

```
device(config)# interface management 0
device(config-if-mgmt-0)#
```

3. Configure the IPv4 or IPv6 address as required.

```
device(config-if-mgmt-0)# [no] ipv4 address
device(config-if-mgmt-0)# [no] ipv46 address
```

4. Configure the IPv4 or IPv6 gateway address as required.

```
device(config-if-mgmt-0)# [no] ipv4 gateway
device(config-if-mgmt-0)# [no] ipv6 gateway
```

MAC Address

MAC address is a unique physical address assigned to a network interface controller (NIC).

A MAC address is represented in a six byte hexadecimal format, xx:xx:xx:xx:xx:xx. The MAC address is divided into the following parts:

- Bits 47–24: Organization Unique Identifier (OUI) assigned by IEEE. (Example: 00:80:2d)
- Bits 23–12: Chassis ID
- Bit 11-9: Type of MAC address in the switch

  If all zeroes (000), it is a port address (physical MAC address); otherwise it is a VLAN address (virtual MAC address)
- Bits 8-0: 512 port MAC addresses
- Bits 11–0: 3584 VLAN MAC addresses

```
47                   24 23            12 11 10 9 8        0

|---------------| |----------| 0  0  0 | Port MAC --|
    IEEE OUI               Chassis     |--VLAN MAC------|
```
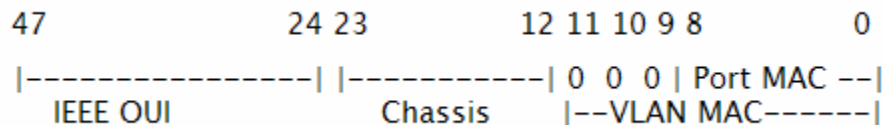
Figure 1: Parts of MAC Address

Ingress

You can configure ingress policies which are associated with ACLs, route-maps, listener policies, and other settings to perform actions to be taken on packets at ingress.

Note the following when configuring an ingress policy:

- An ingress policy can specify only one route-map.
- A route-map bound to the ingress-group can be modified
- An ACL bound to a listener policy can be modified.
- An ACL bound to a listener-policy or route-map can be modified.

Ingress-Group

An ingress-group is used to link tap interfaces to the specified route-map, which defines actions to be taken on packets at ingress.

Create an Ingress Group

Ingress-group classifies the packets received on 9920 at ingress.

Before You Begin
Note the following naming conventions for name identifiers:

- The name cannot exceed 64 characters.
- The name must start with an alphabet character or an underscore.
- The name must contain alphanumeric characters and special characters (underscores, hyphens, or periods).
- The following reserved keywords cannot be used as name identifiers: `all`, `ingress-group`, `egress`, `egress-group`, `match`, `list`, `access-list`, `route-map`, and `listener-policy`.

About This Task

You can associate an ingress-group with a route-map using the **set** command to apply various actions or policies to the traffic at the specified tap interface.

> **Note**
> A port can either be an individual port or a member port-channel.
> A port associated with an ingress-group configuration cannot be configured as a member port-channel until its active configurations are cleared.

Procedure

1. Enter the Config mode.
   ```
   device(config)#
   ```

2. Create an ingress group.
   ```
   device# ingress-group group-name
   device# configure terminal
   device(config)# ingress-group ingress-group noc_traffic
   device(config-ingress-group)# set route-map rmap1
   ```

3. Verify that the ingress group is created.
   ```
   device# show running-config ingress-group
   ingress-group noc_traffic
     set route-map rmap1
   ```

View Ingress-Group Details

Procedure

1. View ingress-group configuration.
   ```
   # show ingress-group [ name | all ]
   ```

2. View ingress-group counters.

```
# show counters ingress-group [ name | all ]
```

3. View running-config ingress-group information.

```
# show running-config ingress-group
```

Replication

Extreme 9920 software replicates network traffic to multiple ports and port-channels.

Extreme 9920 software achieves replication using egress groups. Replication Service is responsible for forwarding the packets to more than one egress ports. Replication ensures that each monitoring tool has access to necessary packets. The egress groups contain more than one egress object during replication. A route-map or ingress policy always points to an egress group and this ensures that moving between unicast and multicast cases does not require any route-map object changes. The egress-group associated with the route-map can be modified to add or remove egress objects.

Extreme 9920 software uses replication block in the hardware to generate copies for each member. Each copy derives a unique internal identifier which is bound to its own egress-policy (listener-policy) and tunnel-encap properties. If the destination is LAG, load balancing is done based on the system LAG hash properties.

Extreme 9920 software supports the following replication functionality:

- Ingress replication of Layer 3 (IPv4 and IPv6) packets
- Replication of VNTAG, VXLAN, and 802.1BR headers on the ingress ports to an egress port or port group.
- Replication of packets to interfaces without changing information in the packet

For information about replication scale limitations, see *Extreme 9920 Software Scale Matrix, 21.2.1.0*.

Configure Replication

Procedure

1. Enter the global configuration mode.

```
device# configure terminal
device (config)#
```

2. Create an egress port.

```
device (config)# egress name
```

3. Associate the egress ports or LAG with the egress.

```
device(config-egress)# egress name
device(config-egress)# precedence number interface port-channel number
```

4. Configure the egress group within the route-map.

```
device(conf)# route-map name permit sequence_number
device(conf-route-map)# set egress-group name
```

5. Add an egress object to the egress-group, pointing to a LAG.

```
device (conf-egress-group)# set egress egress-one
```

Traffic is forwarded to the LAG and load balanced within the LAG.

6. Add or remove LAG members as required for load balancing.

```
device# configure terminal
device(config)# interface ethernet 1/1
device(config-if)# channel-group 1 mode on
device(config-if)# no channel-group 1
```

7. Add another egress object to the egress-group.

```
device (config)# egress-group name
device(conf-egress-group)# set egress egress-two
```

Traffic is replicated and copies are sent to both egress objects.

8. Create the required listener policy.

```
device (config)# listener-policy { name sequence-id }
```

9. Add the required listener policy.

```
device (conf-egress-group)# set listener-policy name
```

The listener policy is applied based on the egress object.

10. Configure the `encap type`.

```
encap-type gre
```

11. Add a new egress object with single port to the egress-group.

```
device (config)# egress-group name
device (conf-egress-group)# egress egress-three
```

This egress object receives plain frames unmodified.

Example

egress toolA

```
# conf
(conf)# egress egressA
(conf-egress)# precedence 10 interface ethernet 0/10
(conf-egress)# set encap encap-100
(conf-egress)# set listener-policy lp-100
```

egress toolB

```
# conf
(conf)# egress egressB
(conf-egress)# precedence 10 interface ethernet 0/20
(conf-egress)# set encap encap-200
(conf-egress)# set listener-policy lp-100
```

egress toolC

```
# conf
(conf)#egress egressC
(conf-egress)# precedence 10 interface ethernet 0/30
(conf-egress)# set encap encap-100
(conf-egress)# set listener-policy lp-200
```

egress-group (replication)

```
# conf
(conf)# egress-group groupABC
```

```
(conf-egress-group)# egress egressA
(conf-egress-group)# egress egressB
(conf-egress-group)# egress egressC
```

route-map

```
(conf)# route-map rmap1 permit 10
(conf-route-map)# match ipv4-acl ipv4_acl1
(conf-route-map)# set egress-group groupABC
```

ingress-group

```
device(config)# ingress-group group-1
device(config-group-1)# set route-map rmap1
```

show running-config

```
device# show running-config egress-group
egress-group groupABC
  egress egressA
  egress egressB
  egress egressC
```

show egress-group

```
device# show egress-group groupABC
    Name : groupABC
     Description : -
          egress : egressA
          egress : egressB
          egress : egressC
```

Header Modification

You can configure settings to prepare and optimize packet headers for forwarding and processing by analysis tools.

Tagging and encapsulation techniques are an established part of networking, but some tagging protocols can create blind spots or unintended packet loss when third-party applications are not designed to interpret some or all of them. Stripping encapsulation header tags removes the interpreting burden and prepares them for downstream tools.

The NPB application provides the following header-modification functionality:

- Tag stripping
- Header-tag addition

For more information about decapsulation, see Tunnel Termination.
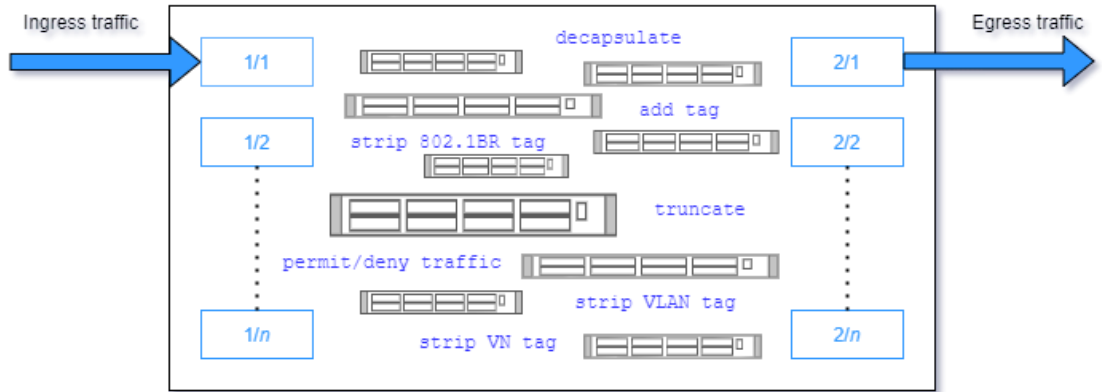
Figure 2: Header-modification and other packet actions

Header-Modification Flow

Header modification actions are performed as specified at ingress and egress.

Based on configuration settings in the route-map or the Access Control List (ACL), when the application receives a packet, it evaluates the packet against the settings, and performs the appropriate actions. Ingress policy actions are applied before the egress policy actions.

Header-modification actions include tag stripping, tag addition, and decapsulation. You specify header-modification actions in a listener policy, which is bound to an egress policy and refers to ACLs and packet-modification settings.
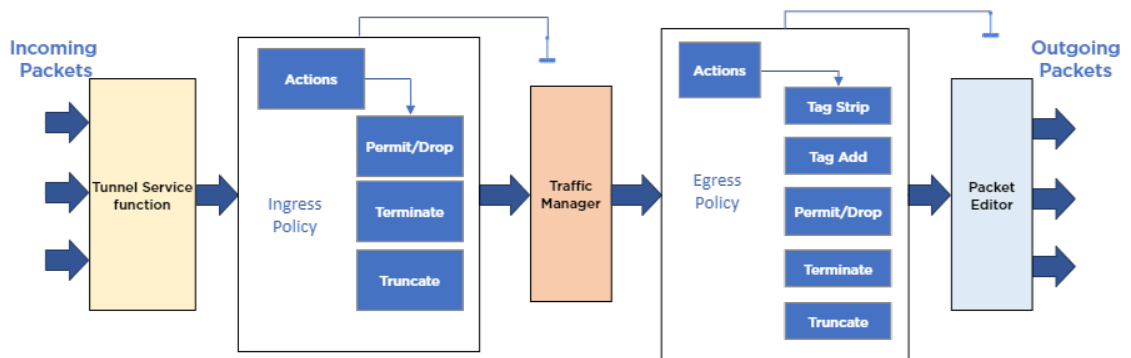
Figure 3: Header and packet modification flow

Tag-Stripping

Encapsulation header tags are stripped to prepare the packets for downstream analytic and other monitoring tools.

Header-tag stripping at egress is applied on only the first L2 header of the received packet.

You can configure a listener policy to define the actions to be performed on packet header tags and bind it to an egress policy. A listener policy supports the following header-modification and traffic actions.

- 802.1BR
- VN-TAG
- VLAN

You can create multiple listener-policy stanzas, but they do not have to be used in unique 1:1 configurations. In the following example, two listener policies (lp-1 and lp-2) are used to manage three traffic types. The listener-policy, `lp-1` has two entries to manage 802.1BR and VN-TAG traffics.

Table 5: Managing traffic types in a listener-policy

Example 1	Example 2	Example 3

`listener-policy lp-1 10`
`match ip access-list`
`v4_someacl1`
`strip br-tag`
	`listener-policy lp-1 20`
	`match ip access-list`
	`v4_someacl2`
	`strip vn-tag`
		`listener-policy lp-2 10`
		`match ip access-list`
		`v4_someacl1`
		`strip vn-tag`

The following table lists the minimum requirements that must be configured for each policy type.

Table 6: Minimum policy configuration requirements

Ingress policy (packet flow)	Egress policy (packet prep for analysis tools)
Match ACL	Match ACL
Route-map with a valid match statement referencing a created ACL.	A designated interface or multiple interfaces.

Note the following when configuring an ingress or egress policy:

- An ingress policy can specify only one route-map policy.
- An egress policy can specify only one listener policy.
- A route map or listener policy can support multiple match-ACLs.
- An ACL bound to the listener policy can be modified.
- A listener policy bound to the egress policy can be modified.
- A route-map bound to the ingress-group can be modified.

802.1BR Tag-Stripping

Learn the high-level process and how a packet appears before and after 802.1BR tag stripping.

Stripping the 802.1BR header tag prepares the packet for forwarding to analysis tools. The 802.1BR header tag is 8 bytes long and positioned after MAC-SA. This header can be followed by an S-Tag, a C-Tag, or both. The following figure shows the structure and

position of 802.1BR header in a frame and its appearance after configuring a listener policy to strip 802.1BR headers.

Figure 4: 802.1BR header before and after tag stripping

Configure 802.1BR Tag-Stripping

Follow this procedure to strip 802.1BR tags to support the encapsulation type expected by your traffic-analysis tools.

About This Task

- The vn-tag cannot be enabled if the br-tag is already enabled in the same listener policy.
- If a tunneled frame has an 802.1BR tag in the outer L2 header, VXLAN, NVGRE, or GTP header-stripping also deletes the 802.1BR tag.

Procedure

1. Enter the Config mode.

   The command line changes to the configuration mode.
   ```
   device(config)#
   ```

2. Configure an ACL of type IPv4, IPv6, or MAC and any actions.

   ```
   device(config)# ip access-list acl5-ipv4
   device(config-ip-acl)# seq 10 permit ip any any count
   ```

   The specified ACL and configured actions are bound to a listener policy.

3. Create the listener policy, including any action subcommands for the policy.

   > **Note**
   > A listener policy supports only one IPv4 ACL, IPv6 ACL, or MAC ACL.

   ```
   device(config)# listener-policy lp-2 24
   device(config-listener-policy)# match ip access-list acl5-ipv4
   device(config-listener-policy)# strip br-tag
   device(config-listener-policy)# description "Strips 802.1BR tags"
   ```

4. Configure an egress policy, and bind the listener policy, specifying any additional egress actions.

> **Note**
> An egress can be associated with only one listener policy.

```
device(config)# egress e2
device(config-egress)# set listener-policy lp-2
device(config-egress)# description DirectTool
device(config-egress)# set encap encap-1
device(config-egress)# precedence 1 interface ethernet 1/14
```

5. Configure an egress group and associate it with the egress policy.

```
device(config)# egress-group eg_1
device(config-egress-group)# description e-group_1
device(config-egress-group)# set egress e2
```

6. Configure the route-map and set any other parameters, such as forwarding actions, match IP access list, and the egress-group.

> **Note**
> A route-map policy supports only one match-ACL per layer.

```
device(config)# route-map R1 10
device(config-route-map)# match ip access-list acl5-ipv4
device(config-route-map)# set egress-group eg_1
device(config-route-map)# forward-action permit
```

7. Configure an ingress group and associate a route map.

> **Note**
> An ingress group can be associated with only one route map.

```
device(config)# ingress-group TAP_TRAFFIC
device(config-ingress-group)# set route-map R1
```

8. Configure the interface port and channel for ingress traffic.

> **Note**
> In the following example, traffic is coming in on slot/port number 2/3.

```
interface ethernet 2/3
description From_TAP
set ingress-group TAP_TRAFFIC
no shutdown
```

9. Configure the interface port and channel for egress traffic.

> **Note**
> In the following example, traffic is leaving on slot/port number 2/14.

```
interface ethernet 2/14
speed 100000
description To_Tool
no shutdown
```

VLAN Header Tag Stripping

Learn the high-level process and how a packet appears before and after VLAN tag stripping.

The VLAN tag is the first tag after MAC-SA and is 4 bytes long. The following figure shows the structure and position of the VLAN header in a frame and its appearance after configuring a policy to strip VLAN headers.
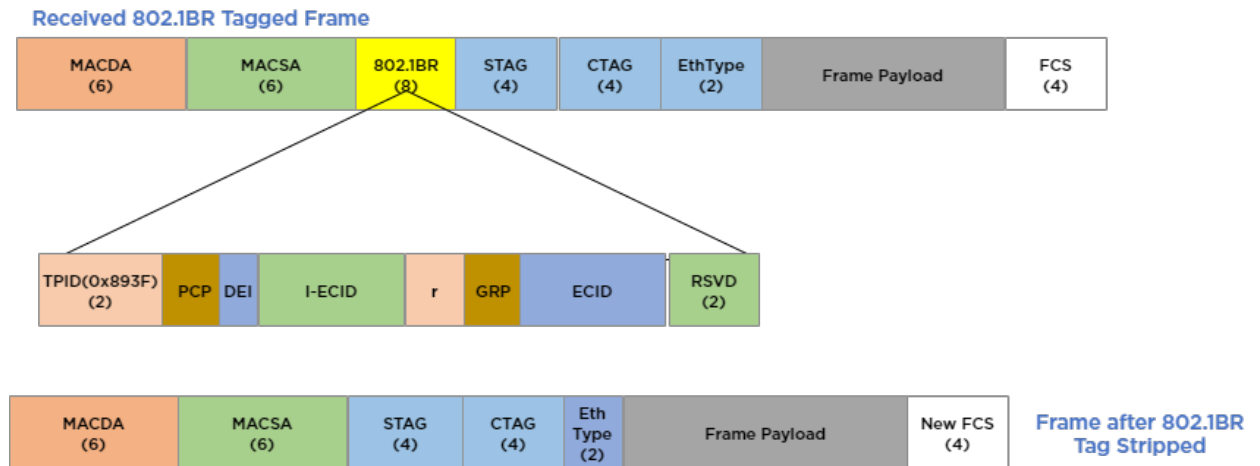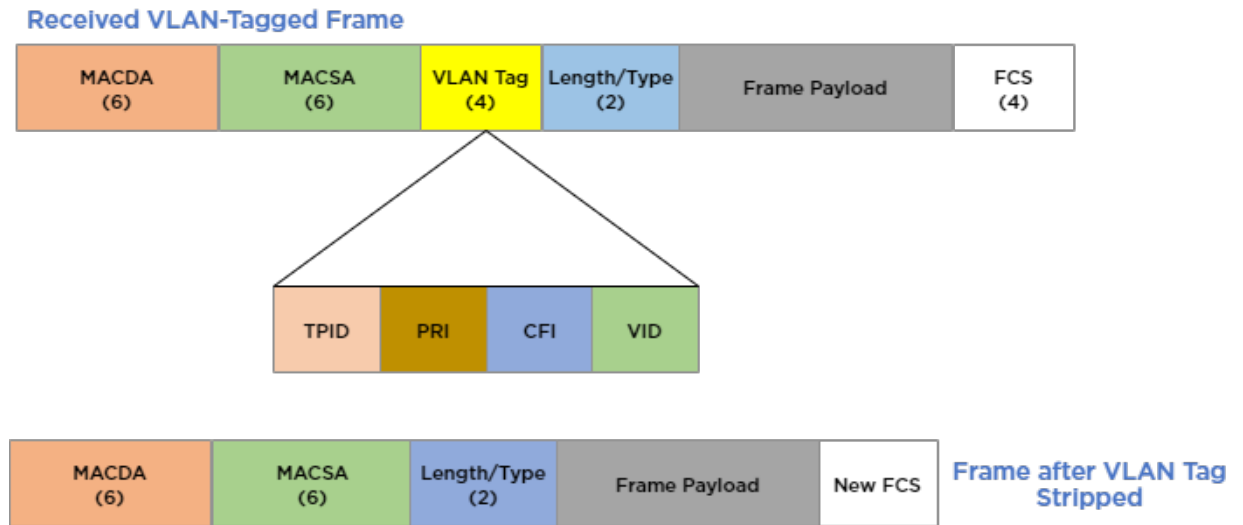

Figure 5: VLAN tag before and after tag stripping

Configure VLAN Tag Stripping

Follow this procedure to strip VLAN tags to support your traffic-analysis tools expected encapsulation type.

About This Task

The VLAN tag is not inside the packet and must be stripped at the ingress device before the encapsulated packet is sent into the network. You can configure VLAN tag stripping so it is performed at ingress or egress.

Procedure

1. Enter the Config mode.

   ```
   device(config)#
   ```

2. Configure an ACL of type IPv4, IPv6, or MAC and any actions.

   ```
   device(config)# ip access-list acl3-ipv4
   device(config-ip-acl)# seq 12 permit ip any any count
   ```

   The specified ACL and configured actions will be bound to a listener policy.

3. Create the listener policy, including any action sub-commands for the policy.

> **Note**
>
> A listener policy supports only one IPv4 ACL, IPv6 ACL, or MAC ACL.

```
device(config)# listener-policy lp-10 5
device(config-listener-policy)# match ip access-list acl3-ipv4
device(config-listener-policy)# strip vlan-tag
device(config-listener-policy)# description "ipv4 listener policy, strip vlan tag"
```

4. Configure an egress policy and bind the listener policy, specifying any additional egress actions.

> **Note**
>
> An egress can be associated with only one listener policy.

```
device(config-egress)# egress e3
device(config-egress)# set listener-policy lp-3
device(config-egress)# description DirectTool
device(config-egress)# precedence 1 interface ethernet 1/14
```

5. Configure an egress group and associate it with the egress policy.

```
device(config)# egress-group eg_5
device(config-egress-group)# description e-group_5
device(config-egress-group)# set egress e3
```

6. Configure the route map and set any other parameters, such as forwarding actions, match ip access list, and the egress-group.

> **Note**
>
> A route-map policy supports only one match-ACL per layer.

```
device(config)# route-map R2 10
device(config-route-map)# match ip access-list acl3-ipv4
device(config-route-map)# set egress-group eg_5
device(config-route-map)# forward-action permit
```

7. Configure an ingress group and associate a route map.

> **Note**
>
> An ingress group can be associated with only one route map.

```
device(config)# ingress-group TAP_TRAFFIC
device(config-ingress-group)# set route-map R2
```

8. Configure the interface port and channel for ingress traffic.

> **Note**
>
> In the following example, traffic is coming in on slot/port number 2/3.

```
interface ethernet 2/3
description From_TAP
ingress-group TAP_TRAFFIC
no shutdown
```

9. Configure the interface port and channel for egress traffic.

> **Note**
> In the following example, traffic is leaving on slot/port number 2/14.

```
interface ethernet 2/14
description To_Tool
no shutdown
```

VN Tag Stripping

Learn the high-level process and how a packet appears before and after VN Tag stripping.

The VN Tag is the first tag after MACSA and is 6 bytes long, It can be followed by an STAG, a CTAG, or both. The following figure shows the structure and position of VN Tag in a frame before and after VN Tag stripping.
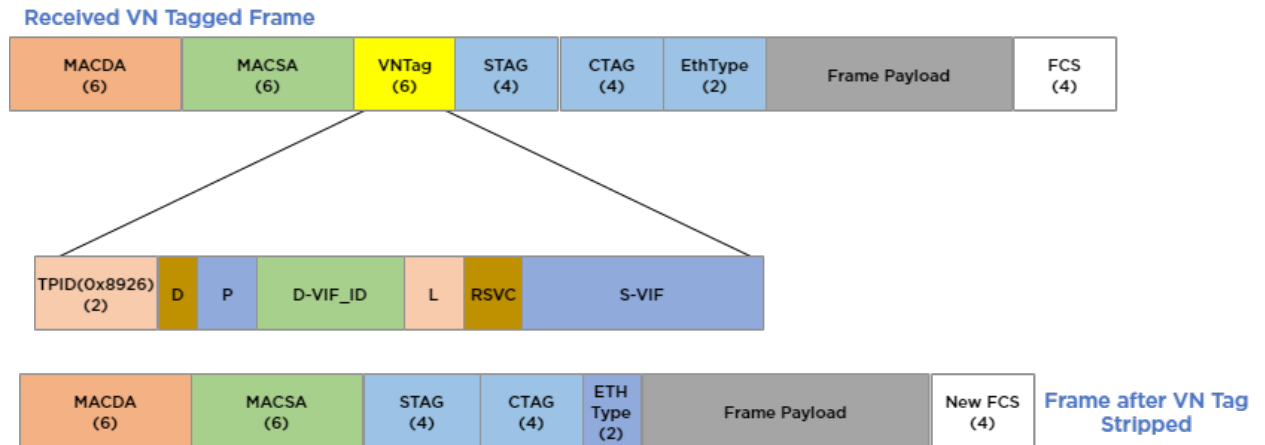
Figure 6: VN tag before and after stripping

Configure VN-Tag Stripping

Follow this procedure to strip VN tags to support the encapsulation type expected by your traffic-analysis tools.

About This Task
The vn-tag cannot be enabled if the br-tag is already enabled in the same listener policy.

Procedure

1. Enter the Config mode.
   ```
   device(config)#
   ```

2. Configure an ACL of type IPv4, IPv6, or MAC and any actions.
   ```
   device(config)# ipv6 access-list acl2-ipv6
   device(config-ipv6-acl)# seq 9 permit ipv6 any any count
   ```
   The specified ACL and configured actions will be bound to a listener policy.

3. Create the listener policy, including any action subcommands for the policy.

> **Note**
> A listener policy supports only one IPv4 ACL, IPv6 ACL, or MAC ACL.

```
device(config)# listener-policy lp-11 43
device(config-listener-policy)# match ipv6 access-list acl2-ipv6
device(config-listener-policy)# strip vn-tag
device(config-listener-policy)# description "LP for vn tag strippin ipv6"
```

4. Configure an egress policy, and bind the listener policy, specifying any additional egress actions.

> **Note**
> An egress can be associated with only one listener policy.

```
device(config)# egress e7
device(config-egress)# set listener-policy lp-11 43
device(config-egress)# precedence 1 interface ethernet 2/14
```

5. Configure an egress group and associate it with the egress policy.

```
device(config)# egress-group eg_9
device(config-egress-group)# description e-group_9
device(config-egress-group)# set egress e7
```

6. Configure the route map and set any other parameters, such as forwarding actions, match ip access list, and the egress-group.

> **Note**
> A route-map policy supports only one match-ACL per layer.

```
device(config)# route-map R4 10
device(config-route-map)# match ipv6 access-list acl2-ipv6
device(config-route-map)# set egress-group eg_9
device(config-route-map)# forward-action permit
```

7. Configure an ingress group and associate a route map.

> **Note**
> An ingress group can be associated with only one route map.

```
device(config)# ingress-group TAP_TRAFFIC_2
device(config-ingress-group)# set route-map R4
```

8. Configure the interface port and channel for ingress traffic.

> **Note**
> In the following example, traffic is coming in on slot/port number 2/3.

```
interface ethernet 2/3
description From_TAP
set ingress-group TAP_TRAFFIC_2
no shutdown
```

9. Configure the interface port and channel for egress traffic.

> **Note**
> In the following example, traffic is leaving on slot/port number 2/14.

```
interface ethernet 2/14
description To_Tool
no shutdown
```

Tag Addition

If configured, the VLAN tag is added to the stripped packet of both tagged and non-tagged packets in the first L2 header. The following figure shows tag addition for a non-tagged packet.
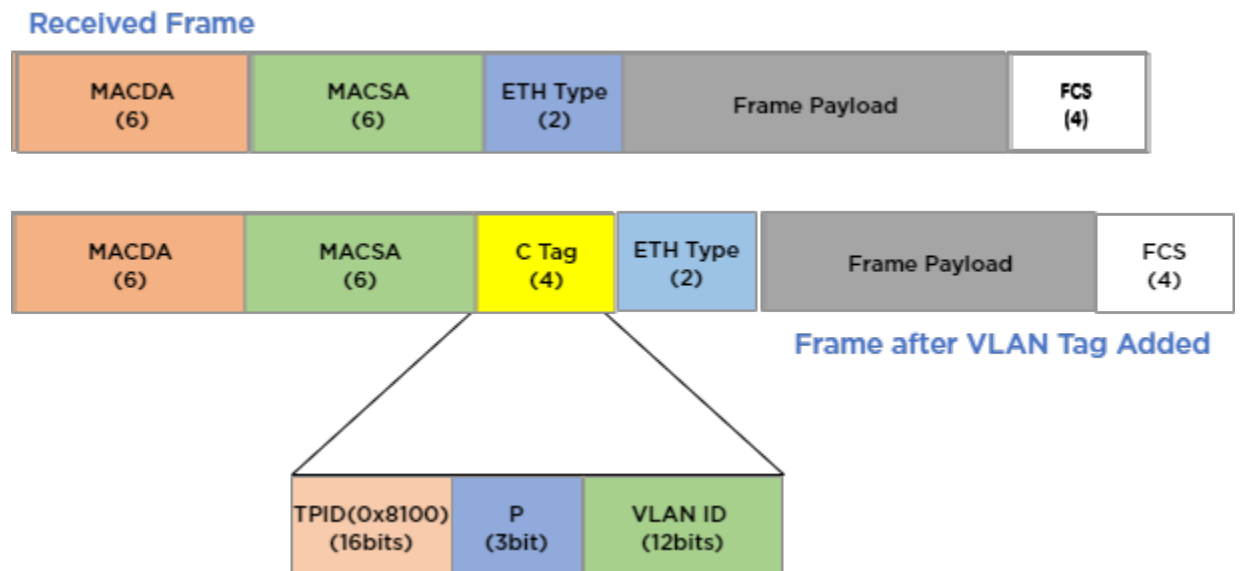
Figure 7: VLAN tag added, non-tagged frames

For a tagged frame, tag-addition is based on the existing tag. The operation is applied to the E Tag, VN Tag, or C Tag, as shown in the following figure. Note the newly added C Tag.

Figure 8: Tag addition, E-tag, VN-tag and C-tag frames

STag and double-CTag additions are no-operation (NOOP) additions, meaning it's a computer processor instruction. The following figure shows an example of tag addition for NOOP cases.
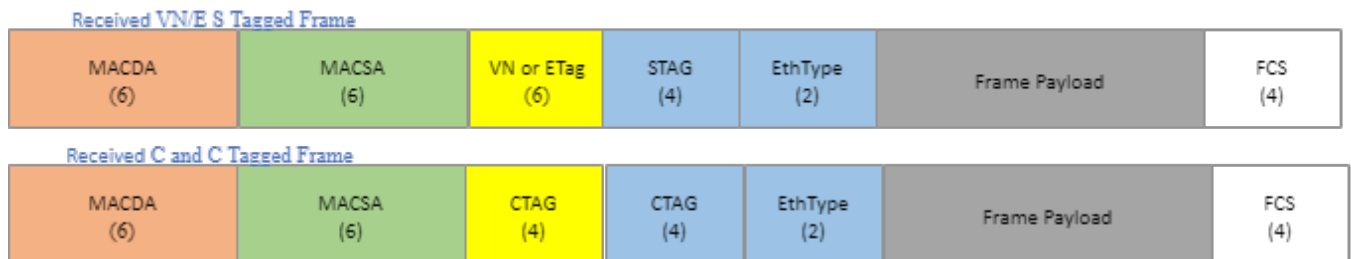
Figure 9: Tag addition, NOOP for tagged frames

Configure Tag Addition

Follow this procedure to add a VLAN tag to tagged or non-tagged packets.

About This Task
Tag addition is performed in the first L2 header.

Procedure

1. Enter the Config mode.

```
device(config)#
```

2. Configure an ACL of type IPv4, IPv6, or MAC and any actions.

```
device(config)# ip access-list acl5-ipv4
device(config-ip-acl)# seq 10 permit ip any any count
```

The specified ACL and configured actions will be bound to a listener policy.

3. Create the listener policy, including any action subcommands for the policy.

> **Note**
> A listener policy supports only one IPv4 ACL, IPv6 ACL, or MAC ACL. The VLAN *id* tag value ranges from 1 to 4095.

```
device(config)# listener-policy lp-2 32
device(config-listener-policy)# match ip access-list acl5-ipv4
device(config-listener-policy)# vlan id
device(config-listener-policy)# description "ipv4 tag add lp"
```

4. Configure an egress policy, and bind the listener policy, specifying any additional egress actions.

> **Note**
> An egress can be associated with only one listener policy.

```
device(config)# egress e2
device(config-egress)# set listener-policy lp-2 32
device(config-egress)# description DirectTool
device(config-egress)# precedence 1 interface ethernet 2/14
```

5. Configure an egress group and associate it with the egress policy.

```
device(config)# egress-group eg_1
device(config-egress-group)# description e-group_1
device(config-egress-group)# set egress e2
```

6. Configure the route map and set any other parameters, such as forwarding actions, match ip access list, and the egress-group.

> **Note**
> A route-map policy supports only one match-ACL per layer.

```
device(config)# route-map R1 10
device(config-route-map)# match ip access-list acl5-ipv4
device(config-route-map)# set egress-group eg_1
device(config-route-map)# forward-action permit
```

7. Configure an ingress group and associate a route map.

> **Note**
> An ingress group can be associated with only one route map.

```
device(config)# ingress-group TAP_TRAFFIC
device(config-ingress-group)# set route-map R1 10
```

8. Configure the interface port and channel for ingress traffic.

> 📝 **Note**
> In the following example, traffic is coming in on slot/port number 2/3.

```
interface ethernet 2/3
description From_TAP
set ingress-group TAP_TRAFFIC
no shutdown
```

9. Configure the interface port and channel for egress traffic.

> 📝 **Note**
> In the following example, traffic is leaving on slot/port number 2/14.

```
interface ethernet 2/14
speed 100000
description To_Tool
no shutdown
```

Packet Truncation and Termination

Some analysis tools do not require all of the packet payload. Packet Truncation removes a portion of the packet length before forwarding at egress.

Tunnel termination removes a tunnel encapsulation header. For more information, see Tunnel Termination.

Packet Truncation Flow

Packet truncation is performed at either ingress or egress before packets are forwarded to analytic tools, depending on the policy configuration. When the incoming packet length is greater than the configured truncation value, the packet is truncated to the length specified in a policy. The following figure is a conceptual representation of packet truncation.
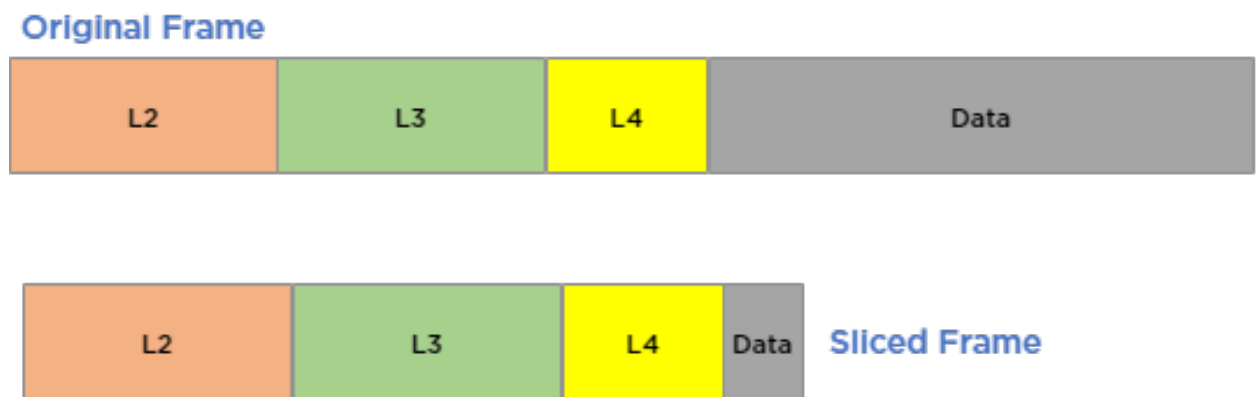
Figure 10: Packet truncation

The incoming packet may be terminated or truncated at ingress as ingress policy actions are applied before the egress policy actions. Egress policy match criteria are

applied to the received tunneled or non-tunneled packets. The order of operations on packets upon ingress to the NPB application is as follows:

1. Tag stripping
2. Tag addition
3. Forward or drop packets
4. Tunnel termination
5. Truncation

Configure Packet Truncation

Follow this procedure to truncate a packet to conform to the length required by your analysis tools.

About This Task
You configure packet truncation in a listener policy. You can include other settings in addition to packet truncation. Minimum truncated packet length is 64 bytes.

Procedure

1. Enter the Config mode.

   The command line changes to the configuration mode.
   ```
   device(config)#
   ```

2. Configure an ACL of type IPv4, IPv6, or MAC and any actions.
   ```
   device(config)# ip access-list acl3-ipv4
   device(config-ip-acl)# seq 12 permit ip any any count
   ```
   The specified ACL and configured actions are bound to a listener policy.

3. Create the listener policy, including the truncate subcommand with any other action subcommands for the policy.

   > **Note**
   > A listener policy supports only one IPv4 ACL, IPv6 ACL, or MAC ACL.

   ```
   device(config)# listener-policy lp-3 18
   device(config-listener-policy)# match ip access-list acl3-ipv4
   device(config-listener-policy)# truncate 512
   device(config-listener-policy)# description "truncate ip packets lp"
   ```

4. Configure an egress policy and bind the listener policy, specifying any additional egress actions.

   > **Note**
   > An egress can be associated with only one listener policy.

   ```
   device(config)# egress e3
   device(config-egress)# set listener-policy lp-3 18
   device(config-egress)# description DirectTool
   device(config-egress)# precedence 1 interface ethernet 2/14
   ```

5. Configure an egress group and associate it with the egress policy.
   ```
   device(config)# egress-group eg_5
   device(config-egress-group)# description e-group_5
   device(config-egress-group)# set egress e3
   ```

6. Configure the route map and set any other parameters, such as forwarding actions, match ip access list, and the egress-group.

> **Note**
> A route-map policy supports only one match-ACL per layer.

```
device(config)# route-map R2 10
device(config-route-map)# match ip access-list acl3-ipv4
device(config-route-map)# set egress-group eg_5
device(config-route-map)# forward-action permit
```

7. Configure an ingress group and associate a route map.

> **Note**
> An ingress group can be associated with only one route map.

```
device(config)# ingress-group TAP_TRAFFIC
device(config-ingress-group)# set route-map R2
```

8. Configure the interface port and channel for ingress traffic.

> **Note**
> In the following example, traffic is coming in on slot/port number 2/3.

```
interface ethernet 2/3
description From_TAP
set ingress-group TAP_TRAFFIC
no shutdown
```

9. Configure the interface port and channel for egress traffic.

> **Note**
> In the following example, traffic is leaving on slot/port number 2/14.

```
interface ethernet 2/14
speed 40000
description To_Tool
no shutdown
```

Tunnel Termination Flow

Tunnel termination can be configured for received L2 or L3 packets.

You can configure settings to apply tunnel termination to received packets, either tunneled (both L2 and L3) or non-tunneled. Tunnel termination is performed at either ingress or egress depending on the policy configuration.

L2 tunnel termination flow is as follows:

1. The outer tunnel of L2 tunneled packets is removed.
2. The current position is shifted to the start of the inner L2 header.
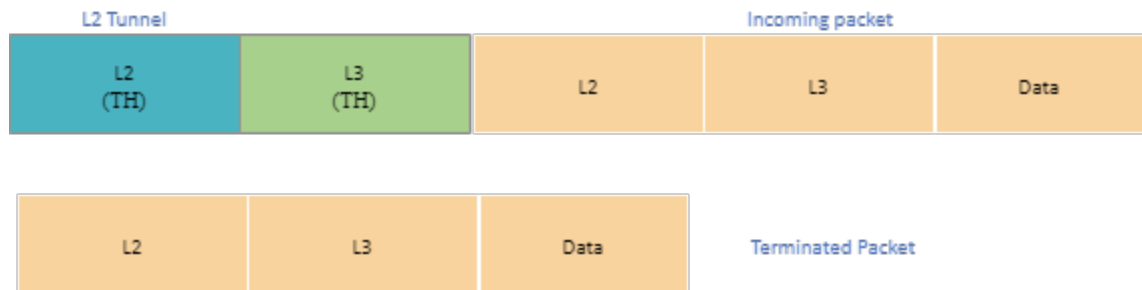
Figure 11: L2 tunnel termination

L3 tunnel termination flow is as follows:

1. The L2 header is retrieved from the L2 outer header because L3 tunneled-packet inner headers do not have the L2 header.
2. The L3 outer header is stripped.
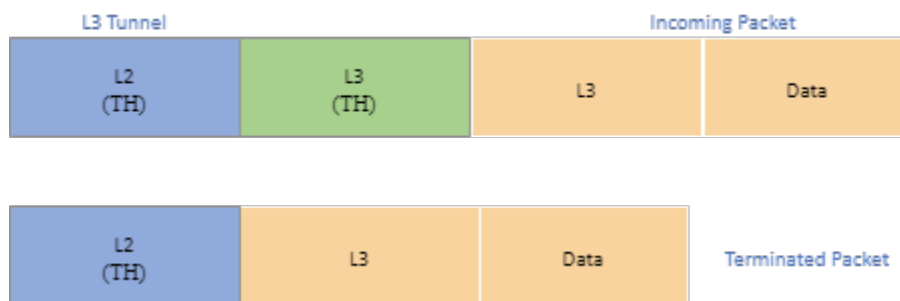
Figure 12: L3 tunnel termination

Forward Incoming Traffic

About This Task
Perform the following procedure to forward incoming traffic using MAC, IPv4, or IPv6 ACL.

Procedure

1. Configure the ingress group with the required ingress ports.
2. Configure L2 (MAC) or L3 ACL (IPv4 or IPv6) in the access list with **Permit** option.
3. Set L2 or L3 ACL in the route-map.
4. Configure the port channel, egress ports, or egress group based on redirect/forward to single egress port or multiple ports in route-map.
5. Apply route-map policy to the ingress group.
6. Send traffic flows to the device under test.
   - The traffic flows or packets are forwarded to egress port or egress group based on L2 or L3 ACL configured in the route-map.
   - Traffic flows are load-balanced between the mapped egress ports.
7. Verify the CLI statistics of ingress group and L2 or L3 ACL counters to determine the number of packets or flows received.

8. Verify the CLI statistics of egress port or egress group to determine the number of packets or flows forwarded.
9. Verify the CLI statistics of L2 or L3 ACL matches the number of packets or flows forwarded.
10. Verify that the packets matching the L2 or L3 ACL are getting forwarded in egress port/egress group by validating or capturing the wired PCAP collected in the analytical tools.
11. Verify the statistics by enabling the logs of forwarding agents.

Drop Incoming Traffic

About This Task
Perform the following procedure to drop incoming traffic using MAC, IPv4, or IPv6 ACL.

Procedure

1. Configure the ingress group with the required ingress ports.
2. Configure L2 (MAC) or L3 ACL (IPv4 or IPv6) in the access list with **Deny** option.
3. Set L2 or L3 ACL in the route-map.
4. Apply route-map policy to the ingress group.
5. Send traffic flows to the device under test.

   The traffic flows or packets are dropped based on L2 or L3 ACL configured in the route-map.
6. Verify the CLI statistics of ingress group and L2 or L3 ACL counters to determine the number of packets or flows dropped.
7. Verify the CLI statistics of egress port or egress group.
8. Verify that the packets matching the L2 or L3 ACL are not getting forwarded in egress port by validating or capturing the wired PCAP collected in the analytical tools.

Filter Message Header

About This Task
Perform this procedure to filter IPGRE, nvGRE, VXLAN, IPIP, or GTPu message headers.

Procedure

1. Configure the ingress group with the required ingress ports.

2. Configure the required IPv4 or IPv6 ACL settings to filter IPGRE, nvGRE, VXLAN, IPIP, or GTPu message headers.

   - Use the L4 port 132 to filter SCTP packets.
   - Starting with release 21.1.1.0, Extreme 9920 software supports outer VXLAN headers.
   - Starting with release 21.1.2.0, Extreme 9920 software supports IPv6 GRE tunnel termination and MPLS Segment Routing in outer tunnel termination.
   - To configure packet mirroring for VXLAN frames, go to step 3. Otherwise, proceed to step 4 on page 49.

3. (Optional) Configure packet mirroring for VXLAN frames.

   a. Enable mirror configuration.

   ```
   device(config)# mirror mirr_1
   device(config-mirror)#
   ```

   b. Configure the mirror destination port in slot/port format.

   ```
   device(config-mirror)# set interface ethernet NAME
   ```

   c. Configure the required VXLAN or MPLS mirror to the ingress-group.

   ```
   device(config-ingress-group)# traffic-type [ vxlan | mpls ] outer mirror mirr_1
   ```

4. Set ACL filtering in the route-map.

5. Apply the route-map policy to the ingress group.

6. Send traffic flows to the device under test.

   - The configured GTP, SCTP, or VXLAN message header is dropped based on ACL configured in the route-map.
   - The non-filtered traffic is forwarded to the egress port/egress group.

7. Verify the CLI statistics of ingress group to determine the number of packets or flows received and dropped.

8. Verify the CLI statistics of egress port or egress group to determine the number of non-filtered packets or flows forwarded.

9. Verify the CLI statistics of UDA ACL matches the number of packets/flows dropped.

10. Verify that the packets matching the UDA ACL are not forwarded in egress groups and only non-filtered packets are received by validating or capturing the wired PCAP collected in the analytical tools.

Filter GTP-Tunneled HTTP Messages

About This Task
Perform this procedure to filter or drop HTTPS traffic frames encapsulated in a version 1 GTP frame based on User Defined Attribute (UDA) ACL policy and allow non-filtered traffic to an EGRESS port or group.

Procedure

1. Configure the ingress group with the required ingress ports.

2. Configure the ACL for filtering GTP tunneled HTTPS messages.

3. Set ACL filtering in the route-map.

4. Apply the route-map policy to the ingress group.
5. Send GTP tunneled HTTPS and HTTP traffic flows to the device under test.

   • The HTTPS traffic tunneled in GTP is dropped based on ACL configured in the route-map.

   • The non-filtered HTTP traffic in GTP is forwarded to the egress port/egress group.

6. Verify the CLI statistics of ingress group to determine the number of packets or flows received and dropped.
7. Verify the CLI statistics of egress port or egress group to determine the number of non-filtered packets or flows forwarded.
8. Verify the CLI statistics of UDA ACL matches the number of GTP tunneled HTTPS packets and flows that are dropped.
9. Verify that the GTP tunneled HTTPS packets matching the UDA ACL are not getting forwarded in egress groups and GTP tunneled HTTP packets are received by validating or capturing the wired PCAP collected in the analytical tools.
10. Verify the statistics by enabling the logs of forwarding agents.

Onboard Packet Capture

Onboard Packet Capture tool allows quick capture of packets received or transmitted on a front panel port. The captured packets are stored in one or more Packet Capture (PCAP) files for debugging.

After capturing the configured number of packets, packet capturing process automatically stops for that particular interface. To save and retrieve packet capture in a PCAP file, packet capture must be stopped globally. Saving packet capture configurations in a config file is not required as packet capture is intended for live debugging for a short period of time only.

The packet capture CLI commands are allowed in the Exec mode. The **show capture packet config** command displays all packet capture configurations on Ethernet ports. The **show capture packet interface** command displays content of the PCAP file when packet capture is in-progress. For more information, see *Extreme 9920 Software Command Reference, 21.2.1.0*.

> **Note**
> • Packet filter based on L2 or L3 is not supported in NPB application.
> • Because packets sent to CPU are rate limited, capturing all Ethernet frames that are received or transmitted on the front panel port is not guaranteed.

PCAP File Management

The packets received from data-path are written to the active PCAP file, `pktcapture_running.pcapng`.

The active PCAP file is renamed and saved as `pktcapture_N.pcapng`, where N is 1-25.

- A maximum of 25 PCAP files with a file size of 100 MB each is supported. Packet capture automatically stops when 25 PCAP files are available. The existing PCAP files have to be removed to restart packet capture.
- The capture writes to the active PCAP file until file size reaches 100 MB. The PCAP file is then renamed and saved.
- If the capture is manually stopped, irrespective of the current file size, the active PCAP file is renamed and saved.

For information about onboard PCAP on EVM, see *Extreme Visibility Manager Administration and User Guide, 6.2.0*.

PCAP File Encoding

The NPB application supports 10 simultaneous packet captures.

PCAP File Decoding

The **show capture packet interface** command shows the active PCAP file, `pktcapture_running.pcapng` in read-only mode. If the active PCAP file is not present, the latest inactive PCAP file information is displayed. For more information on Extreme 9920 software commands, see *Extreme 9920 Software Command Reference, 21.2.1.0*.

> **Note**
> Extreme 9920 software supports multiple CLI agents reading the active PCAP file simultaneously.

Configure Packet Capture

Procedure

1. Enable packet capture on an interface.
   ```
   device# capture packet interface ethernet IFNAME { direction [ both | rx | tx ]
   [packet-count 1-8000 ] }
   ```
2. Start packet capture.
   ```
   device# capture start
   ```
3. Stop packet capture.
   ```
   device# capture stop
   ```

Configure Ingress and Egress ACL Capture

Before You Begin
- Use the onboard PCAP configuration and ACL log option only while debugging.

- All onboard PCAP configurations and ACL logging configuration must be unconfigured after completing debugging.

About This Task

You can use the `log` option to the Access Control Entry (ACE) to capture packet through ACL for IPv4, IPv6, and MAC. All packets forwarded by ACE are captured by onboard packet capture.

Procedure

1. Configure packet capture through ACL.

```
ip access-list route-map-acl
    seq 10 permit ip any 2.2.2.2 255.255.255.255 log
ip access-list listener-policy-acl
    seq 10 permit ip 1.1.1.1 255.255.255.255 any log
```

2. Attach the ACL to a route-map to make it an ingress ACL.

```
route-map route_map_1 1
    forward-action permit
    match ip access-list route-map-acl
```

3. Attach the ACL to a listener policy to make it an egress ACL.

```
listener-policy listener_policy_1 1
    forward-action permit
    match ip access-list listener-policy-acl
```

View Captured Packets

Procedure

1. View captured packets.

```
show capture packet interface ethernet 1/1
```

2. View current capture configuration.

```
show capture packet config
```

3. View metadata of all packet capture files.

```
show capture packet pcapfile-info
```

Example

```
device# show capture packet config
All protocol RX capture is enabled on interface Eth 1/2
All protocol RX capture is enabled on interface Eth 1/3
All protocol RX capture is enabled on interface Eth 1/1
All protocol TX capture is enabled on interface Eth 1/1
```

Tunnel Termination

Tunnel termination classifies and decapsulates incoming packets. Tunnel termination can be done using route-map.

The packets are classified based on configured header parameters and then SAP IDs are assigned. Extreme 9920 software decapsulates packets based on parameters that you configure. There are two types of packet decapsulation:

- Non-transport tunnel termination
- Transport tunnel termination

> **Note**
> - Starting with release 21.1.1.0, Extreme 9920 software supports outer VXLAN headers.
> - Starting with release 21.1.2.0, Extreme 9920 software supports IPv6 GRE tunnel termination and MPLS Segment Routing in outer tunnel termination.

Both transport and non-transport tunnel terminations support IPv4 and IPv6 traffic. Based on the type of incoming traffic and configured tunnel termination parameters, the following actions are carried out:

- Termination
- Scope-shift
- Packet classification

The following configurations are required for successful tunnel termination:

- Encapsulation parameter configurations (L2 and L3 parameters for GRE header)
- Egress group configurations
  - Attaching egress object to egress group
- ACL configurations
- Ingress group configurations (for non-transport tunnel terminations)
- Transport tunnel configurations (for transport tunnel terminations)
- Interface configurations
- Route-map configurations

> **Note**
> Extreme 9920 software, release 21.1.2.0 supports only 32 bits out of 128 bits of IPv6.

Transport Tunnel Termination

Transport tunnel termination applies to packets with destination MAC address within the chassis MAC address and considered "destined to us."

The chassis MAC address is a non-configurable parameter and is configured in the data path at boot-up time.

Transport tunnel termination supports decapsulation of GRE or ERSPAN headers. To terminate destination MAC in the packet:

- The packet is checked for an IPv4 or IPv6 address and type of encapsulation.
- If the source IPv4 or IPv6 address and configured encapsulation type match, an intermediate SAP is assigned to the packet and the packet is marked for dropping.
- The intermediate SAP is matched along with tunnel or encapsulation type and tunnel ID.
- If all the parameters match, it is considered a successful termination and from it, a network SAP is derived.

The packet is marked for decapsulation of transport header, and as the transport header is removed, the scope is automatically shifted to the next header in the packet.

The transport tunnel terminated packets are classified according to non-transport tunnel parameters configured as part of the ingress group. Based on ingress group configurations, the inner header is terminated, scope is shifted to the next header, or packets are classified and SAP is updated in the packet metadata to inner SAP (tunnel SAP). For transport packets, both transport (transport header termination) and non-transport configuration (decapsulate outer header, scope-shift operations, or non-transport tunnel classification) are applied and packet classification is done based on both configurations.

Figure 13: Transport tunnel termination in terminate mode
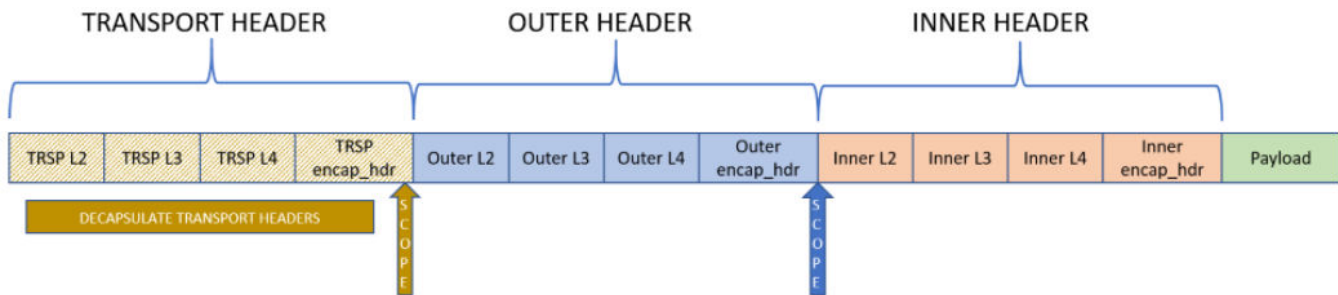
Figure 14: Transport tunnel termination in scope-shift mode
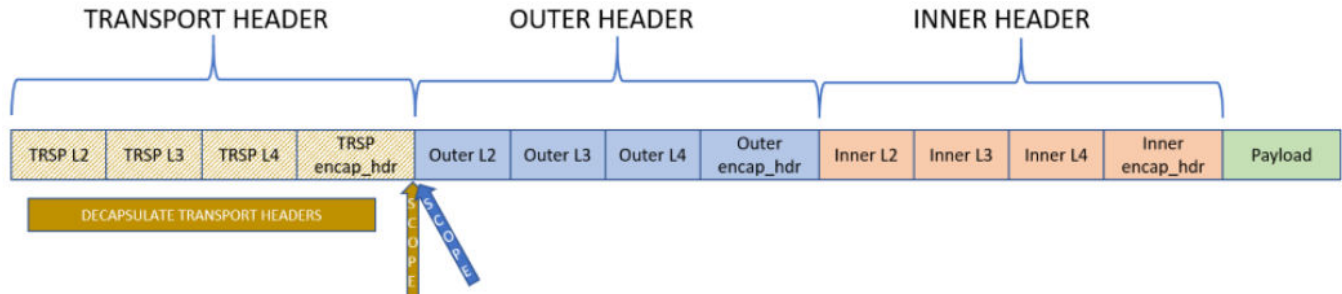
Figure 15: Transport tunnel termination in no-op mode

Non-Transport Tunnel Termination

Non-transport tunnel termination applies to packets that do not have destination MAC address within the chassis MAC address range and considered "not destined to us."

Type of headers to be decapsulated can be configured using CLI or gNMI commands. Non-transport tunnel termination supports header decapsulation for VXLAN, IPGRE, nvGRE, GTP-U, and IP-in-IP.

For VXLAN, GTP-U, and nvGRE packets, a key or identifier can be configured for termination. For example, in VXLAN, a particular VNID can be configured to match for termination. If a key or identifier is not configured for traffic types, all packets are matched irrespective of the key or identifier they ingresses with.

The non-transport tunnel termination parameters can be configured through ingress-group CLI command. Tunnel termination takes the configured action on incoming GRE, GTP, IP-in-IP, NVGRE, and VXLAN packets:

* **Terminate or Decapsulate:** The encapsulation header is decapsulated and the remaining packets are processed in further blocks. This results in shifting the scope of the headers to inner headers automatically.

   Terminate or decapsulate can also be done using route-map or listener-policy.
* **Scope-shift:** There is no header decapsulation and the scope of the header is shifted to inner headers. This results in further blocks in the packet processing pipeline using inner headers of the packet.

   Scope-shift can also be done using route-map.
* **No-operation:** There is no decapsulation or scope shift, only packet classification is done. In ingress group, this results in further processing blocks processing all headers in the packet.
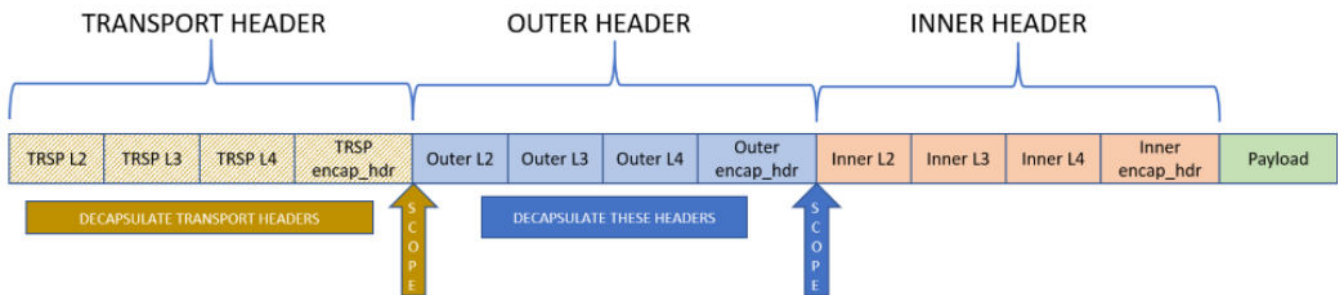
Figure 16: Non-transport tunnel termination in terminate mode
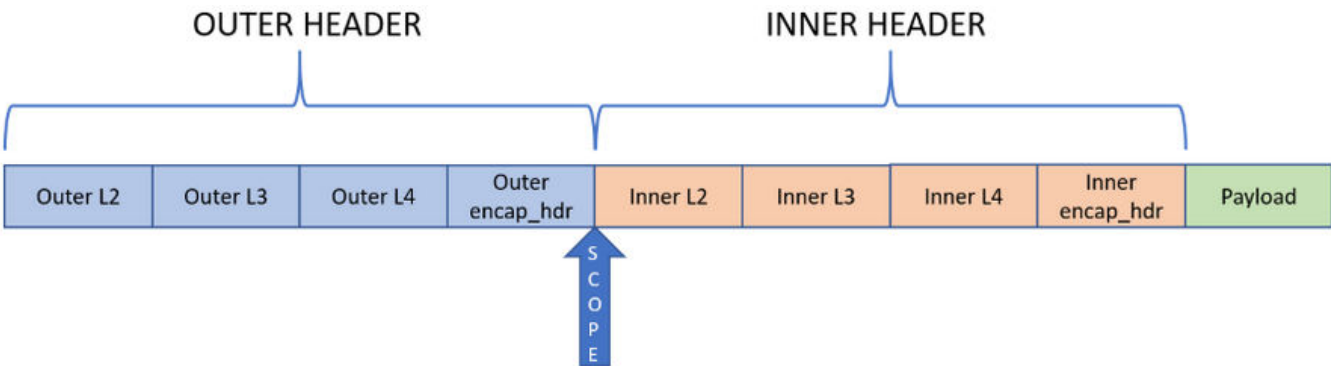
Figure 17: Non-transport tunnel termination in scope-shift mode
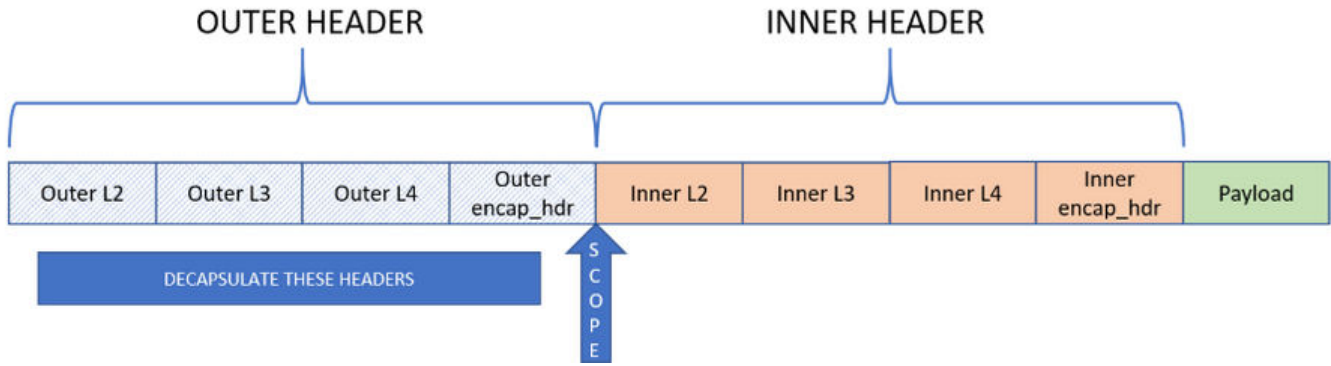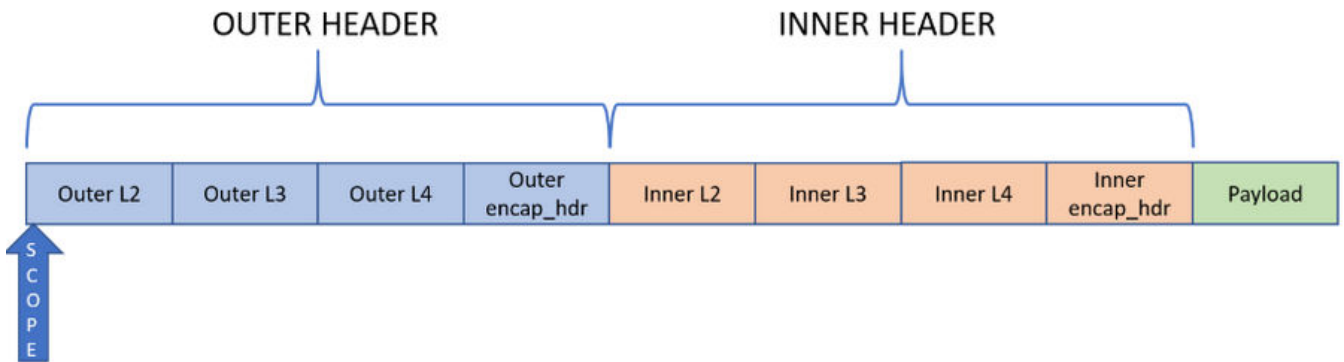

Figure 18: Non-transport tunnel termination in no-op mode

> **Note**
> All VXLAN packets without a VLAN-Tag are considered as Transport tunnel packets even if the transport header destination MAC address does not match the chassis MAC.

Outer VXLAN Header

Starting with release 21.1.1.0, Extreme 9920 software supports outer VXLAN headers. The VXLAN filters can be enabled per port.

The VXLAN frames can be filtered based on VNID, IPv4 source address, and IPv4 destination address.

- VXLAN frames that do not match the filter are dropped.
- VXLAN headers in the matching frames are terminated, and a new SAP is assigned for further processing.
- VXLAN packets without outer tag are processed as transport tunnel packets and the VXLAN headers are terminated.

Limitations

The outer VXLAN header limitations are as follows:

- Multiple ingress-groups can share the same outer VXLAN filter.
- Ingress-groups with the same outer VXLAN filter belong to the same hardware table and share the counter and mirror actions.
- Multiple ingress-groups configured without tunnel ID, source IP address, or destination IP address can create conflicts among ingress-groups.
- The order of priority for ingress frames that match filters from multiple ingress-groups is as follows:

  1. Tunnel ID
  2. Source IP address
  3. Destination IP address

Outer MPLS Header

Starting with release 21.1.2.0, Extreme 9920 software supports outer Multiprotocol Label Switching Segment Routing (MPLS-SR). MPLS-SR packets without an outer VLAN tag (or 802.1BR/VN tag), are terminated if there is a matching configuration present in the ingress-group. The packets are filtered based on MPLS Label ID, Traffic Class, and Time-To-Live values of Bottom Of Stack (BOS) label.

MPLS labels from the outer tunnel are removed, and remaining packets are subjected to inner-tunnel processing or ingress policies. VLANs tagged outer Ethernet packets are not terminated, but are matched at ingress policy stage.

> **Note**
>
> MPLS-SR tunnel termination is supported in the following scenarios:
> - MPLS-SR is the only tunnel present in the packets.
> - MPLS-SR is the outermost tunnel and there is an inner-tunnel (for example, GTPU).

Limitations

The outer MPLS header limitations are as follows:

• The incoming traffic matching is based on the Bottom Of Stack MPLS label values. Other MPLS labels in the stack are not supported.

• The matched MPLS stream is terminated, and all MPLS labels from the label stack are removed.

• A maximum of four MPLS labels are supported.

• The inner frame must have an Ethernet header instead of direct IP header.

• The outer VxLAN tunnel limitations apply to MPLS as well. For more information, see Outer VXLAN Header on page 57.

> **Note**
> GRE tunnel encapsulation does not support MPLS packets. When MPLS packets are subjected to GRE encapsulation, the protocol ID in the GRE header is set to 0.

Packet Mirroring

Packet mirroring mirrors the whole VXLAN frame to another egress port.

For a VXLAN frame without the outer tag, the VXLAN header is terminated and the frame is subjected to further processing, based on ingress and egress configuration.

When mirroring is enabled, one copy of the whole frame is subjected to normal processing where the VXLAN header is terminated and subjected to regular ingress or egress processing.

Another copy of the frame is mirrored with egress port without any header termination.

The filters for VXLAN that are configured using ingress-group can be applied per port. If the frame does not match the filter, it is dropped.

> **Note**
> • Only one mirror destination port is supported.
> • You can use the ingress-group to enable mirroring for outer MPLS-SR termination. MPLS-SR packets that match the filters are sent to the egress port based on the configured mirror.

Enable Mirror Configuration

About This Task
You can use the mirror reference in ingress group to enable mirroring based on filters.

Procedure

1. Enter the Config mode.

```
device# configure terminal
device(config)#
```

2. Enter the mirror configuration mode.

```
device(config)# mirror mirr_1
device(config-mirror)#
```

3. Configure the mirror destination port in slot/port format.

```
device(config-mirror)# set interface ethernet NAME
```

Only one mirror destination port is supported.

Example
The following example shows the configuration of a mirror to an ingress-group:

```
9920(config)# mirror mirr_1
9920(config-mirror)# set interface ethernet 1/10
9920(config-mirror)# do show mirror mirr_1

              Name : mirr_1
       Description : -
         Interface : ethernet 1/10

9920(config-mirror)# ingress-group vv1
9920(config-ingress-group)# traffic-type vxlan outer mirror mirr_1
9920(config-ingress-group)# do show ingress-group vv1

   Name : vv1
           Route-Map : -
         Description : -
          Interfaces : none

  Outer Tunnel Config :
         Traffic-Type : VxLAN
            Tunnel-Id : any
   Destination-ip-addr : any
       Source-ip-addr : any
               Mirror : mirr_1
```

Terminate Transport Tunnels

Before You Begin
- Starting with release 21.1.1.0, Extreme 9920 software supports outer VXLAN headers.
- Starting with release 21.1.2.0, Extreme 9920 software supports IPv6 GRE tunnel termination and MPLS Segment Routing in outer tunnel termination.

About This Task
Perform this procedure to terminate GRE, nvGRE, VXLAN, IPIP, or GTP-U packet headers from incoming packets.

Procedure

1. Create a transport tunnel with the required parameters.

```
device(config)# transport-tunnel name tunnel-type [ gre | erspan ] [ src-ip ipaddr |
mask mask ] [tunnel-id value ]
```

2. Create an ingress-group with the required parameters.

```
device(config)# ingress-group name
```

3. Configure the required ingress-group traffic-type parameters for traffic classification:

   - To configure the traffic-type and mode, go to step 4 on page 60.
   - To configure the traffic-type with the IP address to be matched, go to step 5 on page 60.
   - To configure the VXLAN outermost header, go to step 6 on page 60.
   - To configure the MPLS outer header, go to step 7 on page 60.

4. Configure the required traffic-type parameter and mode for the ingress-group.

   - Terminate mode: Decapsulates IPGRE, nvGRE, VXLAN, IPIP, or GTP-U packet headers.
   - Scope-shift mode: Shifts scope of IPGRE, nvGRE, VXLAN, IPIP, GTP-U packets to inner headers.

   ```
   device (config-ingress-group)# traffic-type {gre | gtpu | ipip | nvgre | vxlan} mode
   [decap | new-scope]
   ```

   - The outer (`sap-id`) and inner (`inner-sap-id`) tunnel SAP IDs are generated for tunnel levels one and two.
   - If the mode to decapsulate or terminate packet headers is not specified, packets are classified based on configured traffic type and tunnel ID parameters.

5. Configure the traffic-type with the IP address to be matched for traffic classification.

   ```
   device(config-ingress-group)# traffic-type
   { gre | ipip | gtpu | vxlan | nvgre } ip [ src-ip src-mask dst-ip dst-mask ]
   ```

6. Configure the required VXLAN outermost header.

   - IP address in double encapsulation traffic:
     ```
     device(config-ingress-group)# traffic-type vxlan
     outer ip src-ip src-mask dst-ip dst-mask
     ```

   - VNI match based classification:
     ```
     device(config-ingress-group)# traffic-type vxlan outer vni vni-value
     ```

   To configure packet mirroring, go to step 8 on page 60. Otherwise, proceed to step 9 on page 61.

7. Configure the MPLS outer header.

   ```
   device(config-ingress-group)# traffic-type mpls outer header1
   { [ label1 | any ] [ tc1 | any ] [ ttl1 | any ] }
   ```

   To configure packet mirroring, go to step 8. Otherwise, proceed to step 9 on page 61.

8. (Optional) Configure the required mirroring for frames.

   a. Enable mirror configuration.

      ```
      device(config)# mirror mirr_1
      device(config-mirror)#
      ```

   b. Configure the mirror destination port in slot and port format.

      ```
      device(config-mirror)# set interface ethernet NAME
      ```

   c. Configure the required VXLAN or MPLS mirror to the ingress-group.

      ```
      device(config-ingress-group)# traffic-type
      [ vxlan | mpls ] outer mirror mirr_1
      ```

9. Attach the ingress group to the required ingress port or interface.

```
device(config-ingress-group)# set ingress-group name
```

10. Attach the ingress group to the transport tunnel.

```
device(config)# transport-tunnel name
device(config-transport-tunnel)# set ingress-group name
```

11. Configure L3-ACL to forward the traffic:

   a. Set ACL in the route-map.
   b. Bind route-map to the ingress-group.

```
device(config)# ip access-list name
device(config)# route-map name sequence_number
device(config-route-map)# match ip name
device(config-route-map-name)# forward-action permit
```

12. Configure the required ports, interfaces, or port-channel:

   • List of egress ports or interfaces in egress-group and associated group in route-map.
   • Egress interface in route-map.
   • Port-channel in egress-group and set the group in route-map.

```
device(config)# interface ethernet slot/port
device(config-if)# set ingress-group ingress-group-name
```

Terminate Non-Transport Tunnels

About This Task
Perform this procedure to terminate non-transport tunnels.

Procedure

1. Create an ingress-group with the required parameters.

```
device(config)# ingress-group name
device (config-ingress-group)#
```

2. Attach the ingress group to the required ingress port/interface.

```
device(config)# interface ethernet 1/12
device(config-if-eth-0/12)# set ingress-group name
```

3. Configure the required ingress-group traffic-type parameters for traffic classification:

   • To configure the traffic-type and mode, go to step 4.
   • To configure the traffic-type with the IP address to be matched, go to step 5 on page 62.

4. Configure the required traffic-type parameter and mode for the ingress-group.
   - Terminate mode: Decapsulates the configured packet headers.
   - Scope-shift mode: Shifts scope of the configured packets to inner headers.

```
device(config-ingress-group)# traffic-type {gre | gtpu | ipip | nvgre | vxlan} mode
[decap | new-scope]
```

   - The outer (`sap-id`) and inner (`inner-sap-id`) tunnel SAP IDs are generated for tunnel levels one and two.
   - If the mode to decapsulate or terminate packet headers is not specified, packets are classified based on configured traffic type and tunnel ID parameters.

5. Configure the traffic-type with the IP address to be matched for traffic classification.

```
device(config-ingress-group)# traffic-type { gre | ipip | gtpu | vxlan | nvgre } ip
[ src-ip src-mask dst-ip dst-mask ]
```

6. Configure the L3-ACL to forward the traffic.
   a. Set the ACL in the route-map.
   b. Bind the route-map to the ingress-group.

```
device(config)# ip access-list name
device(config)# route-map name sequence_number
device(config-route-map)# match {mac | ip | ipv6} name
device(config-route-map-name)# forward-action permit
```

7. Configure the required ports, interfaces, or port-channel:
   - List of egress ports and interfaces in the egress-group and the associated group in route-map.
   - Egress interface in the route-map.
   - Port-channel in the egress-group and the associated group in route-map.

```
device(config)# interface ethernet slot/port
device(config-if)# set ingress-group name
```

View Transport Tunnel Statistics

Procedure

1. View transport tunnel configuration.

```
# show transport-tunnel [ all | name ]
```

2. View transport tunnel counters information.

```
# show counters transport-tunnel name
```

Example

```
# show transport-tunnel tunnel-1
name : tunnel-1
tunnel-type : erspan
tunnel-id : 12345
source IP : 10.10.10.0
source IP mask : 255.255.255.0
dest IP : 20.20.20.0
dest IP mask : 255.0.0.0
ingress-group : ig1

# show counters transport-tunnel tunnel-1
```

```
ERSPAN Terminated Packet Statistics
    RX Frames : 0
     RX Bytes : 0
ERSPAN Dropped Packet Statistics
 Dropped Frames : 0
  Dropped Bytes : 0
```

Terminate and Scope-Shift Matrix

The following table summarizes the behavior of scope-shift and terminate in the following stages of packet processing:

- Ingress Group
- Route Map
- Listener Policy

> **Note**
> - Scope-shift is not supported simultaneously in two stages.
> - If scope-shift comes before a terminate, the terminate action terminates two layers of the tunnel.

Packet Capture

```
Ethernet II, Src: 00:00:00_00:00:02 (00:00:00:00:00:02), Dst: 00:00:00_00:00:01
(00:00:00:00:00:01)
802.1Q Virtual LAN, PRI: 0, DEI: 0, ID: 100
Internet Protocol Version 4, Src: 1.1.1.1, DST: 7.7.7.7          1st GRE Layer
Generic Routing Encapsulation (IP)
Internet Protocol Version 4, Src: 2.2.2.2, DST: 8.8.8.8          2nd GRE Layer
Generic Routing Encapsulation (IP)
Internet Protocol Version 4, Src: 3.3.3.3, DST: 9.9.9.9          3rd GRE Layer
Generic Routing Encapsulation (IP)
Internet Protocol Version 4, Src: 4.4.4.4, DST: 10.10.10.10
Data (886 bytes)
```

#	Ingress Group	Route Map	Listener Policy	Expected Behavior
1	Terminate	Scope-shift	Terminate	• Ingress Group terminates 1st GRE layer. • Route Map lookup happens with 2nd IP layer. • Route map shifts scope to 3rd IP layer. • Listener policy lookup happens with 3rd IP Layer. • 2nd and 3rd GRE layer is terminated by listener policy.
2	Terminate	Scope-shift		• Ingress Group terminates 1st GRE layer. • Route Map lookup happens with 2nd IP layer. • Route map shifts scope to 3rd IP layer. • Listener policy lookup happens with 3rd IP Layer.

#	Ingress Group	Route Map	Listener Policy	Expected Behavior
3	Terminate	Terminate	Terminate	• Ingress Group terminates 1st GRE layer. • Route Map lookup happens with 2nd IP layer. • Route map terminates 2nd GRE layer. • Listener policy lookup happens with 3rd IP Layer. • Listener policy terminates 3rd GRE Layer.
4	Terminate	Terminate		• Ingress Group terminates 1st GRE layer. • Route Map lookup happens with 2nd IP layer. • Route map terminates 2nd IP layer. • Listener policy lookup happens with 3rd IP Layer.
5	Terminate		Terminate	• Ingress Group terminates 1st GRE layer. • Route Map lookup happens with 2nd IP layer. • Listener policy lookup happens with 2nd IP Layer. • Listener policy terminates 2nd GRE Layer.
6	Scope-shift	Scope-shift	Terminate	Scope-shift is not supported simultaneously in two stages.
7	Scope-shift	Scope-shift		Scope-shift is not supported simultaneously in two stages.
8	Scope-shift	Terminate	Terminate	• Ingress Group shifts scope to 2nd IP layer. • Route Map lookup happens with 2nd IP layer. • 1st and 2nd GRE layer is terminated by route map. • Listener policy lookup happens with 3rd IP Layer. • 3rd GRE layer is terminated by listener policy.
9	Scope-shift	Terminate		• Ingress Group shifts scope to 2nd IP layer. • Route Map lookup happens with 2nd IP layer. • 1st and 2nd GRE layer is terminated by route map. • Listener policy lookup happens with 3rd IP Layer.

#	Ingress Group	Route Map	Listener Policy	Expected Behavior
10	Scope-shift		Terminate	• Ingress Group shifts scope to 2nd IP layer. • Route Map lookup happens with 2nd IP layer • Listener policy lookup happens with 2nd IP Layer. • 1st and 2nd GRE layer is terminated by listener policy.
11		Scope-shift	Terminate	• Route Map lookup happens with 1st IP layer. • Route map shifts scope to 2nd IP layer. • Listener policy lookup happens with 2nd IP Layer. • 1st and 2nd GRE layer is terminated by listener policy.
12		Scope-shift		• Route Map lookup happens with 1st IP layer. • Route map shifts scope to 2nd IP layer. • Listener policy lookup happens with 2nd IP Layer. • No change in Packet.
13		Terminate	Terminate	• Route Map lookup happens with 1st IP layer. • 1st GRE layer is terminated by route map. • Listener policy lookup happens with 2nd IP Layer. • 2nd GRE layer is terminated by listener policy.
14		Terminate		• Route Map lookup happens with 1st IP layer. • 1st GRE layer is terminated by route map. • Listener policy lookup happens with 2nd IP Layer.
15			Terminate	• All Lookup happens with 1st IP layer. • 1st GRE layer is terminated by listener policy.
16				• All Lookup happens with 1st IP layer. • No change in egress packet.

Link Aggregation

Link Aggregation (LAG) bundles multiple physical Ethernet links into a single high-bandwidth port-channel for enhanced performance and redundancy.

LAG provides load balancing across physical interfaces and improves reliability. The port-channel stays operational as long as at least one physical interface within the port-channel is operational.

LAG uses various system components such as messaging infrastructure, L2 Protocol Services, Forwarding Services, and Stratum.

Each LAG consists of the following components:

- Links of the same speed
- A MAC address that is different from the MAC addresses of the individual LAG member links.
- An interface index for each link for neighboring devices to identify.
- An administrative key for each link. Only the links with the same administrative key value can be aggregated into a LAG.
- Link Aggregation Control Protocol (LACP) automatically configures an administrative key value equal to the port-channel identification number on each configured link.

> **Note**
> A port can either be an individual port or a member port-channel.
>
> A port associated with an ingress-group or egress configuration cannot be configured as a member port-channel until its active configurations are cleared.

Static LAG

In static link aggregation, you can create a port-channel interface or LAG and add member interfaces manually using the CLI commands or gNMI set requests.

In static link aggregation, Link Aggregation Control Protocol (LACP) packets are not exchanged between the partner systems. Aggregation and load-balancing of frames on static links is determined by the operational status and administrative state of the link.

Min-Links define the operational state of a LAG interface. If the number of operationally up Ethernet ports are less than configured min-links value, the LAG interface is considered operationally down. By default, min-links value is set to 1. At least one member port must be up, for a LAG interface to be operationally up.

> **Note**
> - Member ports in channel-group can be added or deleted without creating the port-channel.
> - Updating state DB and creating hardware entry can be triggered only after creating the port-channel.

Configure Static LAG

About This Task

Perform this procedure to configure a static LAG.

Procedure

1. Enter the global configuration mode and configure the LAG or port-channel with single or multiple interfaces.
   ```
   device# configure terminal
   device(config)# interface port-channel 1
   ```

2. Enter the interface configuration mode and add LAG interfaces or port-channels to ingress (LAG1) and egress (LAG2) groups, as required.

   An interface cannot be part of two LAGs.
   ```
   device(config)# interface ethernet 1/1
   device(config-if)# channel-group LAG1 mode on
   device(config-if)# channel-group LAG2 mode on
   ```

3. Configure the required options for L2 (MAC) or L3 (IP or IPv6) ACL in the access list:
   - L2 (MAC) ACL:
     ```
     # match mac access-list name
     ```
   - L3 (IP) ACL:
     ```
     # match ip access-list name
     ```
   - L3 (IPv6) ACL:
     ```
     # match ipv6 access-list name
     ```

4. Enter the route-map configuration mode and configure L2 or L3 ACL in the route-map.

- Deny: Traffic flows or packets that match this criteria are dropped.
- Permit: Traffic flows or packets that match this criteria are forwarded to egress group or egress port.

```
device# config-route-map
device(config-route-map)# forward-action [ permit | deny ]
```

The traffic flows or packets are dropped or forwarded based on the configured criteria in the route-map.

5. Apply the route-map policy to the ingress group.

```
device (config-ingress-group)# set route-map name
```

6. Send the traffic flows to the device under test.
7. Verify that the traffic flows received from multiple interfaces of LAG1 are aggregated.
8. Verify that the forwarded traffic received at LAG2 is symmetric load balanced across multiple egress ports.
9. Verify the CLI LAG or port-channel statistics to determine the number of flows that are aggregated or load balanced.
10. Verify the CLI ingress or egress group statistics to determine the number of packets or flows dropped.

Link Aggregation Control Protocol

Link Aggregation Control Protocol (LACP) is an IEEE standards-based protocol that allows two partner systems to dynamically negotiate attributes of physical links between them to form logical links.

LACP enables devices to send Link Aggregation Control Protocol Data Units (LACPDUs) to each other and establish link aggregation connections.

The NPB application supports the following LACP functionality:

- Adding or removing LAG or port channel.
- Performing required validations before adding member ports into a LAG.
- Publishing LAG interface status and LAG-IfIndex to State DB.
- Managing LAG member ports.

LACP Modes

LACP operates in two modes:

- Active mode: LACP initiates protocol data unit (PDU) exchanges, regardless of whether the partner system sends LACP PDUs.
- Passive mode: LACP responds to PDUs initiated by its partner system, but does not initiate the LACP PDU exchange.

LACP System Priority

LACP combines the system priority with the MAC address of the switch to form the system ID, and also during negotiation with other switches. The system priority value range is 1 through 65535. The lower numbered system priority values have higher priorities. The default system priority value is 32768.

LACP Timers

LACP timer attributes allow customization of failover time:

- Fast periodic time: LACPDU exchange time is 1 sec and timeout is 3 sec (3 x 1).
- Slow or normal periodic time: LACPDU exchange time is 30 sec and timeout is 90 sec (3 x 30).

The values are set to match the IEEE 802.3ad values by default. The default LACP timer is slow or normal periodic time.

LACP Port Priority

Each port configured to use LACP has an LACP port priority. LACP port priorities range from 1 to 65535. The lower numbered port priority values have higher priorities. LACP combines the port priority and port number to create the port identifier. The port priority is used to determine standby mode ports when there is a hardware limitation that prevents all compatible ports from aggregating. The default port priority value is 32786.

Dynamic LAG

Extreme 9920 software supports Dynamic (LACP) LAG types.

Dynamic LAG uses link aggregation control protocol (LACP) to dynamically aggregate links.

Two partner systems sharing multiple physical Ethernet links can aggregate a number of physical links using LACP. LACP creates a LAG on both partner systems and identifies the LAG by the LAG ID. All links with the same administrative key, and all links that are connected to the same partner switch become members of the LAG. LACP continuously exchanges link aggregation control protocol data units (LACPDUs) to monitor the health of each member link.

Configure Dynamic LAG

About This Task
Perform this procedure to configure a dynamic LAG.

Procedure

1. Enter the interface configuration mode and configure the ingress group with the required ingress ports.

   ```
   device(config)# interface ethernet 1/1
   device(config-if)# ingress-group ingress-group-name
   device(config-ingress-group-name)#
   ```

2. Enter the global configuration mode and configure a LAG or port-channel with single or multiple interfaces.

   ```
   device# configure terminal
   device(config)# interface port-channel 1
   ```

3. Configure LACP.

   ```
   device(config)# protocol lacp
   ```

4. Enter the interface configuration mode and add LAG interfaces or port-channels to ingress (LAG1) and egress (LAG2) groups, as required.

   ```
   device(config)# interface ethernet 1/1
   device(config-if)# channel-group LAG1 mode on
   device(config-if)# channel-group LAG2 mode on
   ```

5. Verify if LACP is able to bring other ports up when the active port is down.

6. Configure the required options for L2 (MAC) or L3 (IPv4/IPv6) ACL in the access list:

   - L2 (MAC) ACL:
     ```
     device# match mac access-list name
     ```

   - L3 (IP) ACL:
     ```
     device# match ip access-list name
     ```

   - L3 (IPv6) ACL:
     ```
     device# match ipv6 access-list name
     ```

7. Enter the route-map configuration mode and configure L2 or L3 ACL in the route-map.

   - Deny: Traffic flows or packets that match this criteria are dropped.
   - Permit: Traffic flows or packets that match this criteria are forwarded to egress group or egress port.

   ```
   device# config-route-map
   device(config-route-map)# forward-action [ permit | deny ]
   ```

   The traffic flows or packets are dropped or forwarded based on the configured criteria in the route-map.

8. Apply route-map policy to the ingress group.

   ```
   device(config-ingress-group-name) # route-map name
   ```

9. Send the traffic flows to the device under test.

10. Verify that the traffic flows received from multiple interfaces of LAG1 are aggregated.

11. Verify that the forwarded traffic received at LAG2 is symmetric load balanced across multiple egress ports.

12. Verify the gNMI or CLI LAG/port-channel statistics to determine the number of flows aggregated and/or load balanced.

13. Verify the gNMI or CLI ingress and/or egress group statistics to determine the number of packets or flows dropped.

14. Verify the statistics by enabling the logs of forwarding agents.

Load Balancing

Based on your policy configuration, the NPB application balances egress traffic load across all active physical interfaces of a port-channel or LAG.
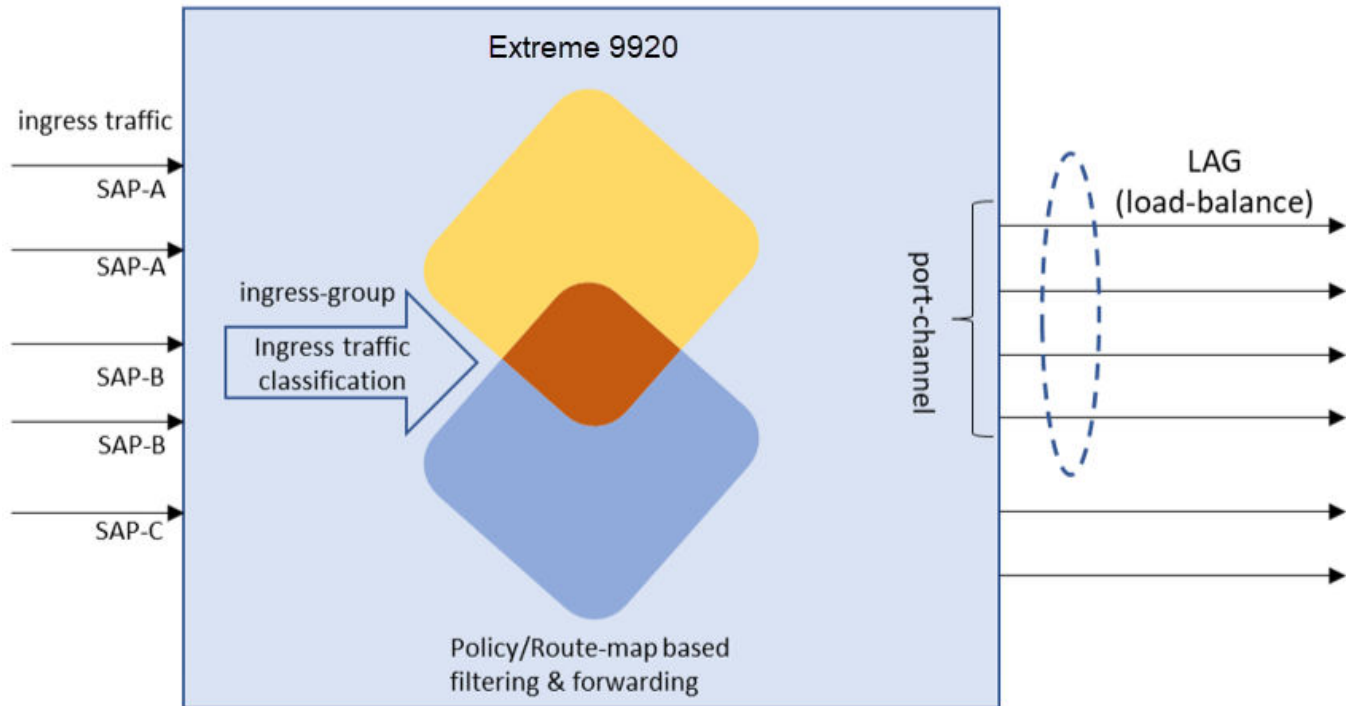
Figure 19: LAG load balancing

> **Note**
> - A physical interface can be part of only one port-channel or LAG.
> - A small drop in traffic can be experienced when an active interface fails.

Enable Load Balancing

About This Task
The NPB application supports GTP tunnel ID based load balancing.

Procedure

1. Enter the interface port-channel configuration mode.

```
Device# configure terminal
Device#(config)# interface port-channel 1
Device(config-if-po-1)#
```

2. Enable load balancing.

```
Device(config-if-po-1)# load-balance src-dst-ip-l4port
```

> **Note**
> For tunnel ID based load balancing, go to step 3.

3. Enable tunnel ID based load balancing.

```
Device(config-if-po-1)# load-balance src-dst-ip-l4port-tid
```

Link Layer Discovery Protocol

Configure LLDP on page 73

IEEE 802.1ab Link Layer Discovery Protocol (LLDP) allows network devices to advertise their identity and capabilities to peer devices on a LAN and store information about the network. The stored network information is used to diagnose and troubleshoot any network issues.

LLDP supports a set of attributes with Type-Length-Value (TLV) to discover neighboring devices. The TLV information provides network mapping, inventory, data, and network troubleshooting information.

Configure LLDP

Procedure

1. Enable LLDP.

   - To enable LLDP globally, perform step 2.
   - To enable LLDP at the interface level, go to step 4 on page 73.

2. Enter the global configuration mode and enable LLDP globally.

   ```
   device# configure terminal
   device(config)# protocol lldp
   ```

3. (Optional) Disable LLDP globally.

   ```
   device(config)# no protocol lldp
   ```

4. Enter the interface configuration mode and enable LLDP at the interface level.

   ```
   device# configure terminal
   device(config)# interface ethernet 0/2
   device(conf-if-eth-0/2)# lldp enable
   ```

5. (Optional) Disable LLDP at the interface level.

   ```
   device(conf-if-eth-1/2)# no lldp enable
   ```

Quality of Service

Quality of Service (QoS) provides preferential treatment to specific traffic that is received on multiple ingress interfaces or Test Access Points (TAPs). QoS aggregates, filters, and forwards traffic to a monitoring tool on an egress interface or an egress group. The forwarding decision is based on the access control lists (ACLs) and route maps applied on the aggregated logical interface or port channel.

QoS allows a networking device prioritize critical traffic steams and provides dedicated bandwidth for effective delivery. In the case of network conjunctions, QoS can selectively drop the low priority streams to allow high priority traffic to pass through. QoS manages traffic delivery using queues, buffers, and schedulers to use the available resources for maximum throughput.

QoS supports eight queues per egress port on a device. The highest queue priority is q7 and q0 (default) is the lowest queue priority. Queue properties cannot be altered, although you can use the `queue` command to change a queue name.

When you configure a route-map, you can attach a QoS forwarding group name and queue priority. The ACL inherits the configuration.

The NPB application uses ACLs to classify and forward packets. An ACL can have multiple access control entries (ACEs). Each ACE has a match criterion and an action. The match criteria are the SAP, IPv4, IPv6, and MAC addresses, VLAN ID, and L4 ports. The actions are to permit or deny, log, and count. When a packet matches the criteria, the specified action is taken for the packets.

Additionally, the ACE includes a QoS forwarding group. You can configure the QoS forwarding group as an optional action in each ACE. If a packet matches the ACE match criteria, the packet is treated with the QoS configurations present in the attached QoS forwarding group.

Several commands support QoS configuration:

- `description`
- `seq (IPv4 access list)`
- `seq (IPv6 access list)`
- `seq (MAC access list)`
- `show running-config qos`

For more information, see:

- Configure a Quality of Service on page 75
- Access Control List on page 14

Configure a Quality of Service

Before You Begin

- Determine whether you are going to attach the QoS forwarding group to a route map or an ACL. ACL provides greater granularity.
- Determine the queue priority to attach.

Procedure

1. Enter the QoS config mode.

```
device# qos
device(config-qos)#
```

2. Create a QoS forwarding group.

```
device(config-qos)# forwarding-group fg1
```

3. Create a queue with the required priority in the format q0 through q7 for the QoS forwarding group.

```
device(config-qos-group)# queue q1
```

- A maximum of eight queues are supported.
- The highest queue priority is q7 and q0 (default) is the lowest queue priority.

4. Attach the QoS forwarding group to the route-map or ACL, as required.

- Attach the QoS forwarding group to the route-map.

```
device(config)# route-map map1 1
forward-action permit
device(config-route-map)# qos-forwarding-group group_name
```

> **Note**
> The route-map allows you to configure actions that are common for all ACLs under the route-map.

- Attach the QoS forwarding group to each access control entry (ACE). If a packet matches the criteria of the ACE, the packets are treated with the QoS configurations present in the attached QoS forwarding group.

```
device (qos-fowarding-group)# ip access-list acl1
seq 10 permit tcp any any qos-forwarding-group fg1
seq 20 permit udp any any qos-forwarding-group fg2
seq 30 permit ip any any
```

> **Note**
> If the forwarding group is not attached, `seq` uses the QoS forwarding group from the route-map for an IPv4 ACL.

SNMP Monitoring

Extreme 9920 software implements Simple Network Management Protocol (SNMP) for monitoring system events. SNMP traps send alert messages from a remote SNMP-enabled device to a central collector. Both IPv4 and IPv6 addresses are supported.

You can configure SNMP to collect the following standard Management Information Base (MIB) objects:

- `interface`
- `ifXTable`
- `system`
- `entPhysicalTable`

Supported SNMP Traps

SNMP traps are alert messages sent from a remote SNMP-enabled device to the SNMP manager.

Extreme 9920 software supports the following generic SNMP traps:

Table 7: Supported SNMP traps

Name	Description	When initialized
coldStart	A coldStart trap signifies that the sending protocol entity is reinitializing itself such that the configuration of the agent or the protocol entity implementation may be altered.	After the device is restarted. After the SNMP service on the device is restarted.
linkDown	A linkDown trap signifies that the sending protocol entity recognizes a failure in one of the communication links represented in the configuration of the agent.	A linkDown trap is activated when the operational status of an interface is changed from UP to DOWN. Example: ```device(config)#
interface ethernet 1/2
device(config-if-
eth-1/2)# shutdown``` The **shutdown** command initializes a linkDown trap.
linkUp	A linkUp trap signifies that the sending protocol entity recognizes that one of the communication links represented in the agent's configuration has come up.	A linkUp trap is activated when the operational status of an interface is changed from DOWN to UP. Example: ```device(config)#
interface ethernet 1/2
device(config-if-
eth-1/2)# no shutdown``` The **no shutdown** command initializes a linkUp trap.

Monitor Events with SNMP

You can use one of the following methods to query data from the NPB application:

- Community string method – Using community string and querying with SNMP v1 or v2c Get or GetNext request.

- User authentication method – Authenticating the SNMP v3 user and then querying with Get or GetNext request to ensure data encryption.

> **Note**
> - Both IPv4 and IPv6 addresses are supported.
> - For information about supported MIBs, see *Extreme 9920 Software MIB Reference, 21.2.1.0*.

Configure the SNMP Community String

You can configure a community string to send notifications of significant system issues and events.

About This Task
Perform this procedure to configure an SNMPv2 community string to access device information through SNMP.

- SNMP access is read-only.
- SNMP v2c is the default version.

Procedure

1. Enter the Config mode.
   ```
   device# configure terminal
   device(config)#
   ```

2. Configure the SNMP v2c community string.
   ```
   device (config)# snmp-server community community-name
   device (config)# snmp-server community extreme03
   ```

   It is important to configure the string correctly. Incorrect string requests are ignored.

3. Exit the configuration mode and verify that the configured SNMP community is listed with all supported communities in the output.
   ```
   device# show snmp server
   snmp-server community extreme03
   ```

Configure an SNMP-Trap Receiver IP Address

You can configure an agent with the SNMP-trap destination information and community.

Before You Begin
This command is available only to users with the admin role.

About This Task
You can enable SNMPv1, SNMPv2c, or SNMPv3 traps for a given host, community, or user combination. A maximum of 12 hosts per community are supported.

Procedure

1. Enter the Config mode.
   ```
   device# configure terminal
   device(config)#
   ```

2. Configure the required SNMP-trap:

- SNMPv1
- SNMPv2c
- SNMPv3

```
snmp-server { host [ ip-address | host ] comm-user udp-port version [ 1 | 2c | 3 ] }
device(config)# snmp-server host 10.23.17.128 public 162 version 2c
```

Domain name, IPv4 address, and IPv6 address are supported as host.

Configure SNMP V3 User

You can configure SNMP v3 users and authenticate them to query the NPB application with Get and GetNext requests. You can reuse the SNMP v3 user configuration to configure SNMP v3 trap.

Before You Begin
Only admin users can perform this procedure.

About This Task
You can use the AES or DES method to encrypt the query data in transit.

Procedure

1. Enter the Config mode.

```
device# configure terminal
device(config)#
```

2. Configure the SNMP v3 user.

```
snmp-server { user [ user name ] auth [ noauth | md5 | sha ] auth-key [ auth-key ]
priv [ nopriv | aes | des ] priv-key [ priv-key ] }
```

Non-admin users have READ-only permissions.

Example

The following example configures an SNMP server user.

```
device(config)# snmp-server user user8 auth sha auth-key authKey1 priv aes priv-key
user1privkey

device(config)# snmp-server user user2 auth md5 auth-key authkey12 priv nopriv

device(config)# snmp-server user user3 auth noauth
```

Platform and Infrastructure Services

The following topics describe platform and infrastructure services.

Chassis Manager

Chassis Manager is responsible for managing hardware modules and maximizing the availability of the services hosted on the chassis.

Interface Manager can subscribe to Chassis Manager to receive and process platform notifications. Management interface supports both IPv4 and IPv6 address configuration.

Stratum

Stratum is an open source silicon-independent switch operating system for software defined networks. It provides P4Runtime APIs for data plane control (programming forwarding behavior and forwarding tables) and Openconfig-based gNMI APIs for configuration and monitoring of switch elements such as ports, Quality of Service (QoS), and platform that are not programmable by P4Runtime. It does not embed any control protocols and is designed to support an embedded or external network operating system.

For more information on gNMI API, see gRPC API Gateway on page 86.

Stratum builds an Openconfig YANG path tree based on the Chassis configuration. Subsequent gNMI `Get`, `Set`, and `Subscribe` requests are supported only for these paths.

For more information on YANG, see the *Extreme 9920 Software YANG Reference, 21.2.1.0*.

Line Card

Chassis Manager detects line card insert or remove events from Open Network Linux Platform (ONLP) and publishes these events along with the line card type and slot number information. The port default properties for the line card type is published in the line card specific configuration file.

Interface Manager updates the chassis configuration file when a line card insert or removal event is received and notifies Stratum of the port map change using GNMI. Stratum processes the difference in the new configuration and initializes the new ports.

Interface Manager is also responsible for updating the operDB with the new interface create events and pushing the configuration for these interfaces down to Stratum. Interface Manager monitors the state of the new interfaces and updates the state to operDB.

> **Note**
> Extreme 9920 software 21.2.1.0 supports 400G speed on the line card.

Configure Line Card

Before You Begin

- Only admin users can perform this procedure.
- Perform this procedure before physically inserting the line card in the slot.
- Power off the line card before removing the slot configuration.
- Only 400G speed is supported on the line card.
- Breakout is not supported for 400G ports.

Procedure

1. Enter the global configuration mode.
   ```
   device# configure terminal
   device(config)#
   ```
2. Configure the slot number and the line card type.
   ```
   device(config)# linecard slot linecard_type
   device(config)# linecard 1 LC16x100G
   ```

   - Valid slot range is 1 through 8.
   - Valid line card type values are LC4x400G and LC16x100G.
3. (Optional) Remove the line card slot configuration.
   ```
   device(config)# no linecard
   ```

Maximum Transmission Unit

Maximum Transmission Unit (MTU) is the size of the largest packet that can be sent over a network. The size of the MTU determines the amount of data that can be transmitted.

If a packet is larger than the MTU allowed by the frame, the device fragments the IP packet into multiple parts that fit into frames, and sends the parts of the fragmented IP

packet separately, in different frames. The device that receives the multiple fragments of the IP packet reassembles the fragments into the original packet. The packet fragmentation can be reduced by increasing the MTU.

Extreme 9920 supports jumbo packets of 9216 bytes, which is the default, and you can change the MTU for individual IP interfaces. The valid range is 1500-9216. The global MTU applies to all interfaces. The interface MTU configuration overrides global MTU configuration.

Running Configuration

NPB application configuration is persistent after device reboot. Upon device reboot, Extreme 9920 is programmed per the persisted configuration.

The configuration currently effective on the device is referred to as the running configuration. Any configuration change you make while the device is online is made to the running configuration.

Apply Configuration

Before You Begin
Before upgrading or downgrading the firmware, back up the running configuration.

Procedure

1. Save the running configuration to a file.

   ```
   device# copy running-config flash://config-file/file-name

   device# copy running-config scp://username:password@host[:port]/filepath

   device# copy running-config usb://file
   ```

2. View configuration files.

   ```
   device# show flash://config-file/filename

   device# show usb://filename
   ```

3. Apply the new configuration from a saved config file.

   ```
   device# copy flash://config-file/file-name running config

   device# copy scp://username:password@host[:port]/filepath running config

   device# copy usb://file running-config
   ```

   Domain name, IPv4 address, and IPv6 address are supported as host.

Reset Configuration

Before You Begin
Before upgrading or downgrading the firmware, back up the running configuration.

Procedure

Save the default configuration to the running configuration to reset device configuration.

```
device# copy default-config running-config
```

Managing Files

Extreme 9920 provides a set of tools for accessing the configuration files. You can use CLI commands to access the flash and USB files.

Table 8: Managing flash files

Command	Task
`dir [flash://[chassis-ms	config-file	coredumps	ifmgr-ms	lacp-ms	lldp-ms	mgmt-cli	mgmt-snmp-agent	mgmtsvc-apigw	ms_images	pcap-file]]	[usb://filename]`	Lists flash and USB directory information.
`show [flash://config-file/ filename]	[usb://filename]`	Displays a flash or USB file.
`delete [flash://config-file/ filename]	[usb://filename]`	Deletes a flash or USB file.

Enable USB Access

Extreme 9920 supports USB storage devices. USB access is disabled by default.

Before You Begin
Only admin users can perform this procedure.

About This Task
USB access is disabled after firmware upgrade, downgrade, or reboot.

Procedure

1. Enable USB access.

   ```
   device# usb enable
   ```

2. (Optional) Disable USB access.

   ```
   device# no usb enable
   ```

Port Management

The NPB application allows port management of the Extreme 9920-NPB-8 and 9920-16C devices.

Configure Breakout Mode

Extreme 9920 supports breakout configuration.

About This Task
The breakout mode enables you to partition a high-speed port into multiple low-speed ports to accommodate multiple data lanes at lower bandwidths.

Procedure

1. Enter the global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the hardware configuration mode.

   ```
   device# (config)# hardware
   device (config-hardware)#
   ```

3. Configure the required breakout mode.

   - `4x10g`: Partition a port into four 10G interfaces
   - `4x25g`: Partition a port into four 25G interfaces

   ```
   device (config-hardware)# breakout [ 4x10g | 4x25g ]
   ```

Port Creation

After a cold reboot or firmware upgrade, Interface Manager waits for the following events to complete to create ports on the system.

- Chassis Manager to detect and publish the chassis type to `operDB`.
- Stratum to initiate configuration management and P4 services, reconfigure ports using the chassis config file, and start Stratum gNMI service.

Interface Manager performs the following tasks to create ports:

1. Connects to stratum and creates interfaces based on the chassis type and configuration.
2. Queries user configurations from state DB and pushes them to stratum.
3. Subscribes to operational state updates from stratum for each of the ports.
4. Updates `operDB` with configuration and state of the ports.

Network Time Protocol

Network Time Protocol (NTP) maintains uniform time across all devices in a network.

NTP is based on a hierarchical model that consists of a local NTP client and remote time servers. The NTP client requests and receives time information from one or

more remote time servers. The local NTP client reviews the time information from all available time servers and synchronizes its internal clock to the most accurate time server.

Two types of time servers exist in the NTP model:

- Primary Time Servers: A primary time server is directly synchronized to a primary reference source, usually a wire or radio clock that is synchronized to a radio station that provides a standard time service. The primary time server is the authoritative time source in the hierarchy, meaning that it is the one true time source to which the other NTP devices in the subnet synchronize their internal clocks.
- Secondary Time Server: A secondary time server uses a primary time server or one or more secondary time servers to synchronize its time, forming a synchronization subnet. A synchronization subnet is a self-organizing, hierarchical master-backup configuration with the primary servers at the root and secondary servers of decreasing accuracy at successive levels.

Date and Time Settings

Extreme devices maintain the current date and time inside a battery-backed real-time clock (RTC) circuit. Date and time are used for logging events. Device operation does not depend on the date and time; a device with incorrect date and time settings can function correctly. However, because the date and time are used for logging, error detection, and troubleshooting, you should set them correctly.

Time Zone Settings

The time zone settings have the following characteristics:

- The setting automatically adjusts for Daylight Saving Time (DST).
- Changing the time zone on a device updates the local time zone setup and is reflected in local time calculations.
- System services that have already started will reflect the time zone changes only after the next reboot.
- Time zone settings are not affected by NTP server synchronization.

The NTP commands support the configuration of an external time server to maintain synchronization among all local clocks in a network.

NTP Server

A Network Time Protocol (NTP) server is used to obtain the correct time from an external time source and adjust the local time in each connected device. When the NTP server functionality is enabled, the NTP server starts listening on the NTP port for client requests and responds with the reference time. Up to eight server addresses in IPv4 or IPv6 format can be configured. When you configure multiple NTP server addresses, the first obtainable address is set as the active NTP server. If there are no reachable time servers, then local device time is the default time until a new active time server is configured.

The NTP server is stateless and does not maintain any NTP client information. Network time synchronization is guaranteed only when a common external time server is used by all devices.

NTP Peer

An NTP peer is a member of a group of NTP servers.

An NTP peer is used to synchronize clock information among a group of servers that serve as mutual backups for each other. If an NTP peer loses all reference clocks or fails, the other peers continue to provide time to other NTP clients on the network.

Configure NTP

After you set the date and time on a device, you can synchronize the local time on a device with a Network Time Protocol (NTP) server.

About This Task

The date and time are configured in the Exec mode and only have to be configured once per device because the value is written to nonvolatile memory. After the basic time information is set up, an NTP server is configured to allow the local time to be synchronized across the network.

Procedure

1. Enable NTP.

   ```
   ntp enable
   ```

2. Disable NTP.

   ```
   no ntp enable
   ```

3. Configure the NTP peer.

   ```
   ntp peer [ ip address | domain name ]
   ```

4. Remove the NTP peer.

   ```
   no ntp peer [ address | domain name ]
   ```

5. Configure the NTP server.

   ```
   ntp server [ address | domain name ]
   ```

6. Remove the NTP server.

   ```
   no ntp server [ address | domain name ]
   ```

gRPC API Gateway

The NPB application supports the Remote Procedure Calls (RPCs) defined in the gRPC Network Management Interface (gNMI) specification. Extreme 9920 supports gNMI version 0.7.0.

The complete text of the gNMI specification, `gnmi-specification.md`, is available on github.com: https://github.com/openconfig/reference/blob/master/rpc/gnmi/gnmi-specification.md.

Extreme 9920 supports the following gRPC Network Operations Interface (gNOI):

- System
- File
- OS
- Interface

Table 9: Supported RPCs

RPC	Purpose	Section in the Specification
`Capabilities`	Used by the client and target as an initial handshake to exchange capability information. Consists of the following messages: • `CapabilityRequest` • `CapabilityResponse`	*Section 3.2 Capability Discovery*
`Get`	Used by the client to retrieve snapshots of the data on the target. Consists of the following messages: • `GetRequest` • `GetResponse`	*Section 3.3 Retrieving Snapshots of State Information*
`Set`	Used by the client to modify the state of the target. Consists of the following messages: • `SetRequest` • `SetResponse`	*Section 3.4 Modifying State*
`Subscribe`	Used by the client to control subscriptions to data on the target. Consists of the following messages: • `SubscribeRequest` • `SubscriptionList` • `SubscribeResponse`	*Section 3.5 Subscribing to Telemetry Updates*

Note

`Get`, `Set`, and `Subscribe` RPCs support both IPv4 and IPv6 addresses.

The RPCs use `openconfig` YANG paths to identify the object on which create, read, update, and delete (CRUD) operations are to be performed. For more information, see the *Extreme 9920 Software YANG Reference, 21.2.1.0*.

Logging

The following topics describe event logging in the NPB application.

Event Logging

Management Services modules use common logging or event subsystem to log various messages and generate events (in standardized forms) to indicate progress, status change, errors, and failures.

All forwarding agents are integrated with the elog feature and support the following category of logs:

- Error
- Warning
- Info
- Debug

The NPB application supports UDP, TCP, and TCP, or TLS transport protocols for remote system logging. For more information, see *Extreme 9920 Software Security Configuration Guide, 21.1.2.0*.

Forward Agent Logs

Based on the event, the forwarding agents print the logs with the required information for debugging and understanding of the scenarios. Also, each log is added with the time stamp and the logs from different agents can be correlated.

The forwarding agents are integrated with the infra tracing support which indicates the functions entry and exit point to ease debugging.

All forwarding agents logs are available at `/var/log/pods/<container-specific-id>/<forwarding-agent-name>/<number>.log` where `<number>` indicates the number of times the container is restarted. The latest or highest-numbered log file must be taken for recent logs.

```
/var/log/pods/ngnpb_agent-sp-intf-ms-85cbc88bcb-cmc9j_eb47ced9-e318-4112-
b5b3-3cf5c61cff7a/agent-sp-intf-ms/25.log
```

```
/var/log/pods/ngnpb_agent-sp-nhop-ms-6db59887d4-xrq9p_429d1d16-7ddf-49d6-89d8-
f7f25e8fce25/agent-sp-nhop-ms/25.log
```

```
{"log":"\u001b[37mTRAC\u001b[0m[06:01:29.588 28-05-2020] addIngressSFPSel:
Enter
\n","stream":"stdout","time":"2020-05-28T06:01:29.58855818Z"}
```

```
{"log":"\u001b[37mTRAC\u001b[0m[06:01:29.588  28-05-2020] addIngressSFPSel:to be added
sfpsel entry {Priority:0  ControlHandle:0 Sfc:[0 13] Result1:[] Result2:[{Valid:true Spi:
[0 0 8]  Si:[6] SiPredec:[6]}]}
\n","stream":"stdout","time":"2020-05-28T06:01:29.58866029Z"}
```

```
{"log":"\u001b[37mTRAC\u001b[0m[06:01:29.600 28-05-2020] addIngressSFPSel:Added sfpsel
entry {Priority:0 ControlHandle:0 Sfc:[0 13] Result1:[]  Result2:[{Valid:true Spi:[0 0 8]
Si:[6] SiPredec:[6]}]}
\n","stream":"stdout","time":"2020-05-28T06:01:29.600646936Z"}
```

```
{"log":"\u001b[37mTRAC\u001b[0m[06:01:29.600 28-05-2020] addIngressSFPSel:
Exit
\n","stream":"stdout","time":"2020-05-28T06:01:29.600774916Z"}
```

Failure Logs

```
{"log":"\u001b[36mINFO\u001b[0m[06:46:39.799 21-05-2020] *** routemap_1 sap12 12
\n","stream":"stdout","time":"2020-05-21T06:46:39.799420893Z"}
{"log":"\u001b[36mINFO\u001b[0m[06:46:39.799 21-05-2020] ***SAP-POLICY ASSOCIATION
routemap_1 sap12 12 \n","stream":"stdout","time":"2020-05-21T06:46:39.79954709Z"}
{"log":"\u001b[36mINFO\u001b[0m[06:46:39.799 21-05-2020] Route
Map Binding route-map:routemap_1 serviceIF:sap12
\n","stream":"stdout","time":"2020-05-21T06:46:39.799740477Z"}
{"log":"\u001b[31mERRO\u001b[0m[06:46:39.803 21-05-2020] Ingress Entry Insert error:
\u0026\{0xc0012ac090 0xc0012aa390 0xc0012bc380 0xc001216580 0xc0012b0540 64 1 0}
; \u0026{66 0 [0 0] [0 0] [0 0 0 0] [0 0 0 0] [0 0 0 0] [0 0 0 0] [9] [255] [0] [0] [0]
[0] [0] [0] [0] [0] [0 0 0 0] [0 0 0 0] [0 12] [255 255] [0] [0] [0 0] [0 0] [0 0] [0 0]
{false}
{false} {true [0] [0] [0] [0] [0 0] [1] [0 13] [0] [0] [0] [0] [0]}};
rpc error: code = PermissionDenied desc = Write from non-master is not
permitted.; \u0026{sap12 12 3 1 true 13} {false 0} {false 0} {false 0}
{false 0} {false 0} {false 0} {false 0} {false 0 {0xc0003b2bc0} 0xc004467680}
\n","stream":"stdout","time":"2020-05-21T06:46:39.803203753Z"}
```

```
{"log":"\u001b[31mERRO\u001b[0m[06:46:39.803 21-05-2020] Bind Acl IPv4-1 serviceIF:12
owner:3 direction:1 Err:1
```