CONFIGURATION GUIDE

Extreme SLX-OS Network Packet Broker Configuration Guide, 18s.1.00

Supporting the ExtremeSwitching SLX 9140 and SLX 9240 Switches

9035270
29 June 2018

Contents

Preface

Document conventions

The document conventions describe text formatting conventions, command syntax conventions, and important notice formats used in Extreme technical documentation.

Notes, cautions, and warnings

Notes, cautions, and warning statements may be used in this document. They are listed in the order of increasing severity of potential hazards.

NOTE
A Note provides a tip, guidance, or advice, emphasizes important information, or provides a reference to related information.

ATTENTION
An Attention statement indicates a stronger note, for example, to alert you when traffic might be interrupted or the device might reboot.

CAUTION
A Caution statement alerts you to situations that can be potentially hazardous to you or cause damage to hardware, firmware, software, or data.

DANGER
A Danger statement indicates conditions or situations that can be potentially lethal or extremely hazardous to you. Safety labels are also attached directly to products to warn of these conditions or situations.

Text formatting conventions

Text formatting conventions such as boldface, italic, or Courier font may be used to highlight specific words or phrases.

Format	Description
bold text	Identifies command names.
	Identifies keywords and operands.
	Identifies the names of GUI elements.
	Identifies text to enter in the GUI.
italic text	Identifies emphasis.
	Identifies variables.
	Identifies document titles.
`Courier font`	Identifies CLI output.

Format	Description
	Identifies command syntax examples.

Command syntax conventions

Bold and italic text identify command syntax components. Delimiters and operators define groupings of parameters and their logical relationships.

Convention	Description
bold text	Identifies command names, keywords, and command options.
italic text	Identifies a variable.
[ ]	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, for example, passwords, are enclosed in angle brackets.
...	Repeat the previous element, for example, *member*[*member*...].
\	Indicates a "soft" line break in command examples. If a backslash separates two lines of a command input, enter the entire command at the prompt without the backslash.

Extreme resources

Visit the Extreme website to locate related documentation for your product and additional Extreme resources.

White papers, data sheets, and the most recent versions of Extreme software and hardware manuals are available at www.extremenetworks.com. Product documentation for all supported releases is available to registered users at www.extremenetworks.com/support/documentation.

Document feedback

Quality is our first concern at Extreme, and we have made every effort to ensure the accuracy and completeness of this document. However, if you find an error or an omission, or you think that a topic needs further development, we want to hear from you.

You can provide feedback in two ways:

- Use our short online feedback form at http://www.extremenetworks.com/documentation-feedback-pdf/
- Email us at internalinfodev@extremenetworks.com

Provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

Contacting Extreme Technical Support

As an Extreme customer, you can contact Extreme Technical Support using one of the following methods: 24x7 online or by telephone. OEM customers should contact their OEM/solution provider.

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

- GTAC (Global Technical Assistance Center) for immediate support
  - Phone: 1-800-998-2408 (toll-free in U.S. and Canada) or +1 408-579-2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact.
  - Email: support@extremenetworks.com. To expedite your message, enter the product name or model number in the subject line.
- GTAC Knowledge - Get on-demand and tested resolutions from the GTAC Knowledgebase, or create a help case if you need more guidance.
- The Hub - A forum for Extreme customers to connect with one another, get questions answered, share ideas and feedback, and get problems solved. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.
- Support Portal - Manage cases, downloads, service contracts, product licensing, and training and certifications.

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

- Your Extreme Networks service contract number and/or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any action(s) 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

About This Document

What's new in this document

Please note the following documentation details for this release:

- This guide is the only publication released with 18s.1.00.
- We assume that you are enabling network packet broker (NPB) on a previously installed 17s.1.02 setup.
- For documentation of legacy, new, and changed commands supporting NPB, refer to Appendix A: NPB Command Reference on page 53.
- If you need other publications, please refer to the 17s.1.02 versions at https://www.extremenetworks.com/support/documentation/, under Routing and Switching > Switch Operating Software > SLX-S Series.

NOTE
On October 30, 2017, Extreme Networks, Inc. acquired the data center networking business from Brocade Communications Systems, Inc. This document has been updated to remove or replace references to Brocade Communications, Inc. with Extreme Networks, Inc., as appropriate.

The following table describes new features in this release:

TABLE 1 Changes in Extreme SLX-OS Network Packet Broker Configuration Guide, 18s.1.00

Feature	Description	Described in
IPv6 lookup-profiles	For flexibility with IPv6 header-elements specified in ACLs, you can configure which half of SIP and DIP addresses are copied into the token header-buffer.	Specifying an IPv6 lookup-profile on page 18
User-defined ACLs (UDAs)	UDAs offer more flexibility for NPB than regular ACLs.	UDAs on page 19
Add an 802.1q VLAN tag	VLAN tagging enables you to mark network traffic for custom processing downstream, such as application-specific filtering on an interconnected packet broker or special handling by the analytics application.	Aggregating traffic from interfaces on page 27 Replicating traffic to multiple interfaces on page 30 Forwarding traffic to a port-channel on page 36
Interface-level header stripping	Enables you to configure different types of header stripping on different interfaces.	Interface-level header stripping on page 40

Supported hardware and software

In those instances in which procedures or parts of procedures documented here apply to some devices but not to others, this guide identifies exactly which devices are supported and which are not.

Although many different software and hardware configurations are tested and supported by Extreme Networks, Inc. for this SLX-OS release, documenting all possible configurations and scenarios is beyond the scope of this document.

The following hardware platforms are supported by this release:

- ExtremeSwitching SLX 9140
- ExtremeSwitching SLX 9240

NOTE

Some of the commands in this document use a slot/port designation. Because the SLX 9140 and the SLX 9240 do not contain line cards, the slot designation must always be "0" (for example, 0/1 for port 1).

Basics of Network Packet Broker

NPB overview

A Network Packet Broker (NPB) provides a collection of monitoring tools with access to traffic across the network.

When you configure and reboot an SLX-OS device into NPB mode, the device can function as a Network Packet Broker.

The SLX-OS NPB implementation includes the following features:

- Aggregation: Traffic on multiple ingress interfaces is aggregated and forwarded on a single egress interface ("many to one") to one monitoring tool.
- Replication: Traffic on a single ingress interface is replicated and forwarded on multiple egress interfaces ("one to many") to multiple monitoring tools.
- Load balancing: Aggregation traffic and replication traffic are load-balanced across the members of an egress port-channel interface. Symmetric load balancing is the only type of load balancing supported.
- Header modification: Header stripping can reduce packet overhead, increasing the efficiency of the security and monitoring tools. The dropping of GPRS Tunneling Protocol (GTP) frames that encapsulate HTTPs packets is also supported.
- Timestamping: With Precision Time Protocol (PTP) accuracy, ingress and egress timestamps aid analysis of congestion and security issues. For details, refer to the "Precision Time Protocol (PTP)" section of the *Extreme SLX-OS Management Configuration Guide for SLX 9140 and SLX 9240*.

The following table indicates which interface types are supported for the aggregation, replication, load balancing, and timestamping features:

TABLE 2 Ingress and egress interface types

Feature	Ingress interfaces	Egress interfaces
Aggregation	Multiple physical, port-channel, or mixed interfaces	One physical or port-channel interface
Replication	One physical or port-channel interface	A TVF domain, including multiple physical, port-channel, or mixed interfaces
Load balancing	Not supported	One port-channel interface
Timestamping	Physical and port-channel interfaces	Physical and port-channel interfaces

Route maps under NPB

A route map is a container for one or more numbered permit or deny stanzas; each stanza contains a sequence of statements.

Evaluation of route maps consists of a list scan, from the lowest-numbered stanza to the highest-numbered stanza. Within each stanza, statements are evaluated in order. The following technologies are some that use route maps:

- BGP and OSPF protocols. For details, refer to the *Extreme SLX-OS Layer 3 Routing Configuration Guide for SLX 9140 and SLX 9240*.
- Policy-based routing (PBR) (not supported)
- Network Packet Broker (NPB)

Route-map syntax and evaluation vary with the technology. However, "match" statements are common to all route-map implementations. Upon a match, the actions specified in the match statement and in the remaining statements of that stanza are implemented. The list scan ends without examining higher-numbered stanzas.

Route-map "set" statements are also supported for the mentioned technologies. Upon a match, set statements perform an action on the matched traffic. The action varies with the technology. The following table compares route-map features and functionality for the various technologies.

TABLE 3 Route-map comparison

Feature	NPB	BGP	PBR (not supported)
Traffic routing or forwarding	Yes	Yes	Yes
Traffic redistribution	No	Yes	No
Route-attribute modification	No	Yes	No
Stanzas	One or more permit or deny stanzas	One or more permit or deny stanzas	One or more permit or deny stanzas
Match statements	One or more **match { mac | ip | ipv4 | uda } address acl** statements	One or more supported **match** statements	One or more **match { ip | ipv4 } address acl** statements
Set statements (supported only in permit stanzas)	**set interface** or **set next-hop-tvf-domain**	0, 1, or multiple **set** statements	0, 1, or multiple **set** statements
Continue statements	No	0 or 1 **continue** statements	No
Interfaces	Physical and port-channel interfaces	Device-level	Layer 3 interfaces

NPB configuration guidelines

Follow these guidelines when implementing Network Packet Broker (NPB):

- NPB requires a license. For details, refer to the *Extreme SLX-OS Software Licensing Guide for SLX 9140 and SLX 9240*.
- A system reboot is required when moving from default system mode to NPB mode and conversely.
- Most Layer 2 and Layer 3 configurations are not supported in NPB mode. Although implementing them does not generate error messages, do not configure them in NPB mode.

- Only one route map can be applied per interface as NPB policy. However, that route map can have multiple stanzas.
- You can apply a given route map on multiple interfaces.
- The only supported match statements are **match { mac | ip | ipv6 | uda } address acl** statements. Other match statements are ignored.
- The only supported set statements are **set interface** and **set next-hop-tvf-domain**. Other set statements are ignored.
- If a stanza does not contain a match statement, "match any" is implied.
- The only supported egress interfaces are physical interfaces, port-channel interfaces, and TVF domains.
- If a physical interface is part of a port-channel, a **set interface** statement on that interface is ignored.

Forwarding priority guidelines

The following guidelines determine forwarding priority in an NPB route-map:

- In general, forwarding priority is determined by the first match in the lowest-numbered stanza. For exceptions, refer to Priority among L2 and L3 match acl statements on page 18.
- The order in which the egress interfaces are configured is also the order for configuring forwarding.
- If a route map has multiple egress interfaces, the first egress interface ready for forwarding is used.
  - A physical interface is considered ready for forwarding if its link is up.
  - Port-channels and TVF domains are considered ready if at least one member port link is up.
- If the current egress interface loses its Ready status, then the next configured egress interface that is ready for forwarding is used.
- If a Down egress interface becomes Ready and if it has higher priority than the currently selected egress interface, the higher priority egress interface replaces it.
- If none of the configured egress interfaces are ready, the traffic is dropped.

Setting NPB as the system mode

Before enabling Network Packet Broker (NPB), you must set NPB as the system mode.

Accessing NPB mode requires a license. For details, refer to the *Extreme SLX-OS Software Licensing Guide for SLX 9140 and SLX 9240*.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter **hardware** to access hardware configuration mode.

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

3. Enter **system-mode npb** to change the system mode to NPB.

   ```
   device(config-hardware)# system-mode npb
   %Warning: To activate the new system-mode config, please reboot the system using 'reload system'.
   ```

   To return to default system mode from NPB system mode, enter **system-mode default** rather than **system-mode npb**.

4. Enter **end** to access privileged EXEC mode.

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

5. To reboot the system, effecting the system mode change, enter **reload system**.

```
device# reload system
Warning: This operation will cause the chassis to reboot and requires all existing telnet,
secure telnet and SSH sessions to be restarted.
Unsaved configuration will be lost.
Please run `copy running-config startup-config` to save the current configuration if not done
already.
Are you sure you want to reboot the chassis [y/n]?
```

6. Press **y** and then **Enter**.

ACLs and UDAs under NPB

Stanza and ACL permit and deny keywords

Both route-map stanzas and access-control lists (ACLs) have **permit** and **deny** keywords.

In NPB mode, you use ACLs only within route-maps, to exclude certain traffic flows from set statements.

Both NPB and PBR route-maps contain **match { mac | ip | ipv4 } address acl** statements. (NPB route-maps can also contain **match uda address acl** statements.) However, permit and deny rules in ACLs applied to route maps function differently than rules in the security ACLs discussed in the *Extreme SLX-OS Security Configuration Guide for SLX 9140 and SLX 9240*:

- In security ACLs, permit rules allow packets and deny rules drop packets.
- In ACLs applied to PBR route-maps, permit and deny rules specify criteria for route-map decisions.
- In ACLs applied to NPB route-maps, permit and deny rules are mechanisms to exclude certain traffic flows from set statements.

The following table describes the interactions between route-map permit and deny stanzas; and permit and deny rules in ACLs applied to those stanzas by **match { mac | ip | ipv4 | uda } address acl** statements.

TABLE 4 Route-map stanza and ACL permit and deny interactions

Stanza	ACL rule	Resulting TCAM action
Permit	Permit	The **set** statement or statements are applied.
Permit	Deny	Packets that match a **deny** keyword are denied from using the stanza **set** statement: • NPB: The packet is dropped. • PBR: The packet is routed as normal.
Deny	Permit	No action is taken: • NPB: The packet is dropped. • PBR: The packet is routed as normal.
Deny	Deny	No action is taken: • NPB: The packet is dropped. • PBR: The packet is routed as normal.

Priority among L2 and L3 match acl statements

In general, forwarding priority is determined by the first match in the lowest-numbered stanza. However, there are exceptions to this priority, if a route map contains match statements from multiple groups:

- Layer 2: **match mac address acl**
- Layer 3: **match { ip | ipv6 } acl**
- User-defined ACLs: **match uda acl**

Priority among Layer 2, Layer 3, and UDA **match acl** statements is as follows:

- If the match within one ACL is on a permit rule and the match in another ACL is on a deny rule, the permit match is accepted and the deny match is ignored.
- If there are multiple Layer 3 matches, the first match in the lowest-numbered stanza is accepted and other matches are ignored.
- If multiple matches—in different ACL types—are on permit rules, only the match in the highest-preference type is implemented; lower-preference matches are ignored. The preference order is Layer 3 > Layer 2 > UDA.

Creating ACLs for NPB

For NPB traffic aggregation and traffic replication, an access-control list (ACL) or user-defined ACL (UDA)—included in a route-map—is required.

This task creates an ACL. If needed, refer also to Creating a UDA on page 24.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **{ mac | ip | ipv6 } access-list** command to create an ACL.

```
device(conf)# ip access-list standard aclNPB_01
```

3. Create one or more permit or deny rules.

```
device(conf-ipacl-std)# permit host 192.1.1.1 count
```

NOTE
A deny rule specifies that a matching packet is denied from using the **set** statement. For details, refer to NPB overview on page 13.

Specifying an IPv6 lookup-profile

For flexibility with IPv6 header-elements specified in ACLs, you can configure which half of SIP and DIP addresses are copied into the token header-buffer.

Each forwarded frame has a token with a 100-byte header buffer for storing header data. Fields copied into this token header-buffer are available for lookup (to make forwarding decisions). Because of this small buffer size, copying the entire 128-bit SIP and DIP addresses is not supported. You can configure the IPv6 lookup profile, specifying which half of SIP and DIP addresses are copied into the buffer:

- (Default) Host ID (bits 64-127)
- Network ID (bits 0-63)

Configuration guidelines:

- The software does not prevent you from specifying a full 128-bit IPV6 address while configuring an IPv6 ACL. However, only the half configured in the IPv6 lookup profile is matched.
- Before you change lookup-profile, you need to evaluate the impact of the change on current IPv6 ACLs.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter **hardware** to access hardware configuration mode.

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

3. Enter the **profile ipv6-lookup** command to specify the IPv6 address lookup-mode.
   - To change from the default Host ID mode to the Network ID mode, enter **profile ipv6-lookup network-id**.

```
device(config-hardware)# profile ipv6-lookup network-id
```

   - To restore the default Host ID mode, enter **profile ipv6-lookup default**.

```
device(config-hardware)# profile ipv6-lookup default
```

UDAs

User-defined ACLs (UDAs) offer more flexibility for NPB than regular ACLs.

In NPB, there are flows that regular ACLs cannot filter. Packets in such flows require filtering based on deep packet inspection or on a combination of MAC and IP fields. UDAs—also known as Flex ACLs—parse deeper and more flexibly into packets for the needed filtering,

UDA implementation flow

A UDA starts functioning on an interface—for aggregation, replication, or forwarding—only if the following flow is implemented:

- Create a UDA profile, using the **uda-key profile** command.
- For the profile, specify the header types in the expected packet structure, using the **flow** command.
- For the profile, specify the header fields to match, using the **uda-key** command.
- Apply the profile to the interface, using the **uda-profile-apply** command.
- Create a UDA, using the **uda access-list** command.
- Create one or more permit or deny rules in the UDA, using the [**seq** *seq-value*] { **deny** | **permit** } command.
- Create a route-map, using the **route-map** command.
- Apply the UDA to the route-map, using the **match uda address acl** command.
- In the route-map, specify the egress interface, using the **set interface** or **set next-hop-tvf-domain** command.
- On a physical or port-channel interface, apply the route map to the ingress interface, using the **npb policy route-map** command.

Packet header-fields for UDAs

The following tables display the header fields supported for each packet header-type.

> **NOTE**
> For implementation details, refer to Configuring a UDA profile on page 23.

TABLE 5 ETHERNET (Ethernet header)

Field	Width (Bits)	Description
HAS_INNER_TAG	1	Packet has inner tag?
HAS_OUTER_TAG	1	Packet has outer tag?
HAS_8021BR_TAG	1	Packet has 802.1BR tag?
HAS_VNTAG	1	Packet has VN tag?
ETHER_TYPE	16	EtherType
INNER_VID	16	PCP or DEI or VLAN ID
OUTER_VID	16	PCP or DEI or VLAN ID
8021BR_TAG	32	802.1BR tag
VNTAG	32	VN tag
MAC_SA_16_47	32	Bits 16–47 of SA MAC
MAC_SA_0_15	16	Bits 0–15 of SA MAC
MAC_DA_32_47	16	Bits 32–47 of DA MAC
MAC_DA_0_31	32	Bits 0–31 of DA MAC

TABLE 6 TUN_ETHERNET (Inner Ethernet header in tunneled frames)

Field	Width (Bits)	Description
HAS_INNER_TAG	1	Packet has inner tag?
HAS_OUTER_TAG	1	Packet has outer tag?
ETHER_TYPE	16	EtherType
INNER_VID	16	PCP or DEI or VLAN ID
OUTER_VID	16	PCP or DEI or VLAN ID
MAC_SA_16_47	32	Bits 16–47 of SA MAC
MAC_SA_0_15	16	Bits 0–15 of SA MAC
MAC_DA_32_47	16	Bits 32–47 of DA MAC
MAC_DA_0_31	32	Bits 0–31 of DA MAC

TABLE 7 IPV4 (IPv4 header)

Field	Width (Bits)	Description
DIP	32	Destination IP
SIP	32	Source IP
PROTOCOL	8	IP protocol
TOTAL_LENGTH	16	Total length
TOS	8	Type of service (DSCP & ECN)
ECN	2	ECN
DSCP	6	DSCP

TABLE 8 IPV6 (IPv6 header)

Field	Width (Bits)	Description
NEXT_HEADER	8	Next header
TOTAL_LENGTH	16	Total length
TRAFFIC_CLASS	8	Traffic class (DSCP & ECN)
ECN	2	ECN
DSCP	6	DSCP
DIP1	32	Bits 32–63 or 96–127 of Destination IP
DIP0	32	Bits 0–31 or 64–95 of Destination IP
SIP1	32	Bits 32–63 or 96–127 of Source IP
SIP0	32	Bits 0–31 or 64–95 of Source IP

TABLE 9 ARP (ARP header)

Field	Width (Bits)	Description
TARGET_IP	32	Target IP
TARGET_HW_ADDR_16_47	32	Bits 16–47 of target HW address
TARGET_HW_ADDR_0_15	16	Bits 0–15 of target HW address
SENDER_IP_16_31	16	Bits 16–31 of sender IP
SENDER_IP_0_15	16	Bits 0–15 of sender IP
SENDER_HW_ADDR_32_47	16	Bits 32–47 of sender HW address
SENDER_HW_ADDR_0_31	32	Bits 0–31 of sender HW address

TABLE 10 TCP (TCP header)

Field	Width (Bits)	Description
FLAGS	8	TCP flags
PORTS	32	Source and destination port
DST_PORT	16	Destination port
SRC_PORT	16	Source port

TABLE 11 UDP (UDP header)

Field	Width (Bits)	Description
PORTS	32	Source and destination port
DST_PORT	16	Destination port
SRC_PORT	16	Source port

TABLE 12 SCTP (SCTP header)

Field	Width (Bits)	Description
PORTS	32	Source and destination port
DST_PORT	16	Destination port
SRC_PORT	16	Source port

TABLE 13 IGMP (IGMP header)

Field	Width (Bits)	Description
TYPE	8	Type

TABLE 14 ICMP (ICMP header)

Field	Width (Bits)	Description
TYPE_CODE	16	ICMP type and code

TABLE 15 ICMPV6 (ICMPv6 header)

Field	Width (Bits)	Description
TYPE_CODE	16	ICMPv6 type and code

TABLE 16 MPLS (MPLS header)

Field	Width (Bits)	Description
LABEL0	24	Label 0 (Label, EXP & S-Bit)
LABEL1	24	Label 1 (Label, EXP & S-Bit)
LABEL2	24	Label 2 (Label, EXP & S-Bit)
LABEL3	24	Label 3 (Label, EXP & S-Bit)

TABLE 17 GRE (GRE header)

Field	Width (Bits)	Description
IS_NVGRE	1	Packet has NVGRE encapsulation?
IS_ERSPAN	1	Packet has ERSPAN encapsulation?
IS_L2	1	Is payload L2?
IS_IPV4	1	Is payload IPv4?
IS_IPV6	1	Is payload IPv4?
FLAGS	8	First byte of the GRE header
KEY_24_31	8	Bits 24–31 of GRE key
KEY_0_23	24	Bits 0–23 of GRE key
TNI	24	NVGRE tenant network ID

TABLE 18 VXLAN (VXLAN header)

Field	Width (Bits)	Description
VNI	24	VXLAN network identifier

TABLE 19 GTP (GTP header)

Field	Width (Bits)	Description
TEID	32	GTP tunnel endpoint ID

TABLE 20 PAYLOAD4 (4 bytes)

Field	Width (Bits)	Description
WORD0	32	Word 0 in the payload

TABLE 21 PAYLOAD8 (8 bytes)

Field	Width (Bits)	Description
WORD0	32	Word 0 in the payload
WORD1	32	Word 1 in the payload

TABLE 22 PAYLOAD16 (16 bytes)

Field	Width (Bits)	Description
WORD0	32	Word 0 in the payload
WORD1	32	Word 1 in the payload
WORD2	32	Word 2 in the payload
WORD3	32	Word 3 in the payload

TABLE 23 PAYLOAD32 (32 bytes)

Field	Width (Bits)	Description
WORD0	32	Word 0 in the payload
WORD1	32	Word 1 in the payload
WORD2	32	Word 2 in the payload
WORD3	32	Word 3 in the payload
WORD4	32	Word 4 in the payload
WORD5	32	Word 5 in the payload
WORD6	32	Word 6 in the payload
WORD7	32	Word 7 in the payload

Headers larger than 128 bytes

If a frame exceeds the 128-byte limit, parsing is limited to the header elements under the limit.

Consider a frame with format ETH > IPv6 > UDP > GTP > IPv6 > TCP > payload. Depending on which tags are in the ETH header, such a frame varies from 130 through 150 bytes.

If the 128-byte parsing limit is exceeded, the frame is parsed as IPv6 > GTP > IPv6 instead of IPv6 > GTP > IPv6 > TCP.

Configuring a UDA profile

Use this task to create a UDA profile and define its directives.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **uda-key profile** command to create a UDA profile.

```
device(config)# uda-key profile prof_01
```

3. Enter the **flow** command to define the header types of the expected packet structure.

```
device(conf-uda-profile-prof_01)# flow header0 ETHERNET header1 IPV4 header2 UDP header3 PAYLOAD32
```

4. Enter the **uda-key** commands to assign header fields to UDA keys.

```
device(conf-uda-profile-prof_01)# uda-key0 header0 ETHER_TYPE
device(conf-uda-profile-prof_01)# uda-key1 header1 PROTOCOL
device(conf-uda-profile-prof_01)# uda-key2 header2 DST_PORT
device(conf-uda-profile-prof_01)# uda-key3 header3 WORD1
```

The following example creates a UDA profile and defines its directives.

```
device# configure terminal
device(config)# uda-key profile prof_01
device(conf-uda-profile-prof_01)# flow header0 ETHERNET header1 IPV4 header2 UDP header3 PAYLOAD32
device(conf-uda-profile-prof_01)# uda-key0 header0 ETHER_TYPE
device(conf-uda-profile-prof_01)# uda-key1 header1 PROTOCOL
device(conf-uda-profile-prof_01)# uda-key2 header2 DST_PORT
device(conf-uda-profile-prof_01)# uda-key3 header3 WORD1
```

Applying a UDA profile on a physical interface

Use this task to apply a user-defined ACL (UDA)-profile on an Ethernet interface.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command—specifying the slot and port—to access interface configuration mode.

```
device(config)# interface ethernet 0/2
```

3. Enter the **uda-profile-apply** command—specifying the UDA profile—to apply the UDA profile to the interface.

```
device(conf-if-eth-0/2)# uda-profile-apply prof_01
```

Applying a UDA profile on a port-channel interface

Use this task to apply a user-defined ACL (UDA)-profile on a port-channel interface.

1. Enter the **configure terminal** command to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface port-channel** command, specifying the port-channel number.

```
device(config)# interface port-channel 10
```

3. Enter the **uda-profile-apply** command, specifying the UDA profile.

```
device(config-Port-channel-10)# uda-profile-apply prof_02
```

Creating a UDA

Use this task to create a user-defined ACL (UDA) and define permit and deny rules within.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **uda access-list** command to create the access list.

```
device(config)# uda access-list extended uda_01
```

3. Enter rules, specifying the needed parameters.

```
device(conf-uda-acl-ext)# permit 0x0800 0xFFFF 17 0xFF 3400 0xFFFF 0x11223344 0xFFFFFFFF
```

The following example creates a UDA and defines a permit rule, with statistics enabled for the rule.

```
device# configure terminal
device(config)# uda access-list extended uda_01
device(conf-uda-acl-ext)# permit 0x112233 0xFFFFFF 0x0a0a0001 0xFFFFFFFF 0x0a0b0001 0xFFFFFFFF 0x11223344
0xFFFFFFFF
```

UDA examples

The following examples illustrate the interplay between UDA profile-definitions and UDA rules.

Header flow: ETH > IPv4 > UDP > RAW payload

Matches:

- IPv4-UDP DPORT = 3400
- Second word (4 bytes) of the payload (following UDP)

```
device# configure terminal
device(config)# uda-key profile fkp1
device(conf-uda-profile-fkp1)# flow header0 ETHERNET header1 IPV4 header2 UDP header3 PAYLOAD32
device(conf-uda-profile-fkp1)# uda-key0 header0 ETHER_TYPE
device(conf-uda-profile-fkp1)# uda-key1 header1 PROTOCOL
device(conf-uda-profile-fkp1)# uda-key2 header2 DST_PORT
device(conf-uda-profile-fkp1)# uda-key3 header3 WORD1
device(conf-if-eth-0/1)# uda-profile-apply fkp1
device(config)# uda access-list extended uda_acl_1
device(conf-uda-acl-ext)# permit 0x0800 0xFFFF 0x11 0xFF 0xD48 0xFFFF 0x11223344 0xFFFFFFFF
```

Header flow: ETH > IPv4 > UDP > VXLAN

Match: VXLAN VNID

```
device# configure terminal
device(config)# uda-key profile fkp1
device(conf-uda-profile-fkp1)# flow header0 ETHERNET header1 IPV4 header2 UDP header3 VXLAN
device(conf-uda-profile-fkp1)# uda-key0 header0 ETHER_TYPE
device(conf-uda-profile-fkp1)# uda-key1 header1 PROTOCOL
device(conf-uda-profile-fkp1)# uda-key2 header2 DST_PORT
device(conf-uda-profile-fkp1)# uda-key3 header3 VNI
device(conf-if-eth-0/1)# uda-profile-apply fkp1
device(config)# uda access-list extended uda_acl_2
device(conf-uda-acl-ext)# permit 0x0800 0xFFFF 0x11 0xFF 0x12B5 0xFFFF 0x1F4 0xffffff
```

Header flow: ETH > IPv6 > GRE > IPv4 > TCP > RAW payload (eHRPD)

Match: TCP DPORT = 443

```
device# configure terminal
device(config)# uda-key profile fkp2
device(conf-uda-profile-fkp2)# flow header0 ETHERNET header1 IPV6 header2 GRE header3 IPV4 header4 TCP
device(conf-uda-profile-fkp2)# uda-key0 header4 DST_PORT
device(conf-if-eth-0/2)# uda-profile-apply fkp2
```

```
device(config)# uda access-list extended uda_acl_3
device(conf-uda-acl-ext)# permit 0x1BB 0xFFFF 0 0 0 0 0 0
```

ETH > IPv4 > UDP > VxLAN > ETH > IPv4 > TCP > RAW Payload (VXLAN with IPv4 passenger)

Matches:

- VNI
- Inner IPv4 SIP and DIP

```
device# configure terminal
device(config)# uda-key profile fkp3
device(conf-uda-profile-fkp3)# flow header0 ETHERNET header1 IPV4 header2 UDP header3 VXLAN header4
TUN_ETHERNET header5 IPV4
device(conf-uda-profile-fkp3)# uda-key0 header3 VNI
device(conf-uda-profile-fkp3)# uda-key1 header5 SIP
device(conf-uda-profile-fkp3)# uda-key2 header5 DIP
device(conf-if-eth-0/3)# uda-profile-apply fkp3
device(config)# uda access-list extended uda_acl_4
device(conf-uda-acl-ext)# permit 0x112233 0xFFFFFF 0x0a0a0001 0xFFFFFFFF 0x0a0b0001 0xFFFFFFFF 0 0
```

Traffic Aggregation and Replication

Aggregating traffic from interfaces

This task aggregates and forwards traffic from multiple interfaces to a single, physical egress interface.

1. Configure a regular ACL or a user-defined ACL (UDA):
   - For a regular ACL, refer to Creating ACLs for NPB on page 18.

     ```
     device(config)# ip access-list standard aclNPB_01
     device(conf-ipacl-std)# permit host 192.1.1.1 count
     device(conf-ipacl-std)# exit
     ```

   - For a UDA, refer to UDAs on page 19.
2. Configure a route map that contains the relevant **match { mac | ip | ipv6 | uda } address acl** command.

   ```
   device(config)# route-map npb_map1 permit 1
   device(config-route-map-npb_map1/permit/1)# match ip address acl acl_2
   ```

3. Specify the route-map egress physical interface.
   - Without stripping 802.1q VLAN tags:

     ```
     device(config-route-map-npb_map1/permit/1)# set interface ethernet 0/5
     ```

   - Stripping 802.1q VLAN tags:

     ```
     device(config-route-map-npb_map1/permit/1)# set interface ethernet 0/5 strip-vlan outer
     ```

   - Adding a 802.1q VLAN tag (optionally, in addition to **strip-vlan outer**):

     ```
     device(config-route-map-npb_map1/permit/1)# set interface ethernet 0/5 add-vlan outer 2
     ```

4. Exit to global configuration mode.

   ```
   device(config-route-map-npb_map1/permit/1)# exit
   ```

5. Apply the route map to the ingress interfaces.
   - Physical interfaces

     ```
     device(config)# interface ethernet 0/2
     device(conf-if-eth-0/2)# npb policy route-map npb_map1
     ```

   - Port-channel interfaces

     ```
     device(config)# interface port-channel 10
     device(config-Port-channel-10)# npb policy route-map npb_map1
     ```

The following example configures ingress traffic from Ethernet 0/1 and port-channel 100 to egress Ethernet 0/5.

```
device# configure terminal
device(config)# route-map npb_map permit 10
device(config-route-map-npb_map/permit/10)# match ip address acl acl_2
device(config-route-map-npb_map/permit/10)# set interface ethernet 0/5
device(config-route-map-npb_map/permit/10)# exit
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# npb policy route-map npb_map
device(conf-if-eth-0/1)# exit
device(config)# interface port-channel 100
device(config-Port-channel-100)# npb policy route-map npb_map
```

Replication to multiple interfaces

Traffic on an ingress interface is replicated and forwarded on multiple egress interfaces, usually to multiple monitoring tools.

The replication ingress-interface is one physical or port-channel interface.

Multiple physical and port-channel egress interfaces are supported. However, you first need to associate such interfaces with a Transparent VLAN flooding (TVF) domain. You then use a route map to designate that TVF domain as egress. Ingress traffic is replicated, and forwarded by way of the TVF to the egress interfaces that you specified.

Transparent VLAN flooding (TVF)

TVF forwards packets without any form of CPU intervention, including MAC learning and MAC destination lookups.

This implementation of TVF has the following attributes:

- Traffic is distributed in hardware to all members of the TVF domain.
- Because this feature does not use any MAC address entries in the CAM, it is useful when MAC address entries need to be conserved.
- You can create as many as 4096 TVF domains.
- Domain members can be tagged and untagged ports. There is no software limitation on the number of member ports.
- You can mix and match ports with different speeds.
- At the TVF domain level, load balancing does not occur. If you need load balancing, associate a port-channel with the TVF domain, which balances the loads among the interfaces included in the port-channel.
- CPU intervention is not required, enabling line-rate traffic forwarding.

Creating TVF domains

This task creates one or more Transparent VLAN flooding (TVF) domains.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **tvf-domain** command, specifying one of the valid formats.
   - To create one TVF domain, specify an integer from 1 through 4096.

     ```
     device(config)# tvf-domain 10
     ```

   - To specify a range of TVF domains, insert a hyphen (-) between the beginning and ending integers.

     ```
     device(config)# tvf-domain 20-30
     ```

   - To specify individual domains and ranges of domains, separate them with commas (for example, 1,5-7,55).

     ```
     device(config)# tvf-domain 1,5-7,55
     ```

3. To add a description of a TVF domain, enter the **description** command.

   ```
   device(config)# tvf-domain 10
   device(config-tvf-domain-10)# description Sample TVF description
   ```

4. Enter **exit** to return to global configuration mode.

   ```
   device(config-tvf-domain-10)# exit
   device(config)#
   ```

Assigning a TVF domain to a physical egress interface

This task assigns a TVF domain to a physical egress interface.

1. Enter **configure terminal** to access global configuration mode.

   ```
   device# configure terminal
   ```

2. Enter the **interface ethernet** command to access interface configuration mode.

   ```
   device(config)# interface ethernet 0/5
   ```

3. Enter the **tvf-domain** command, specifying one of the valid formats.
   - To assign one TVF domain to the interface, specify its integer ID.

     ```
     device(conf-if-eth-0/5)# tvf-domain add 10
     ```

   - To assign a range of TVF domains to the interface, insert a hyphen (-) between the beginning and ending integers.

     ```
     device(conf-if-eth-0/5)# tvf-domain add 20-30
     ```

   - To assign individual domains and ranges of domains, separate them with commas (for example, 1,5-7,55).

     ```
     device(conf-if-eth-0/5)# tvf-domain add 1,5-7,55
     ```

   - To assign all defined TVF domains to the interface, enter **tvf-domain all**.

     ```
     device(conf-if-eth-0/5)# tvf-domain all
     ```

   - To assign all defined TVF domains to the interface—except for those specified—enter the **tvf-domain except** option.

     ```
     device(conf-if-eth-0/5)# tvf-domain except 1,2,4-7
     ```

Assigning a TVF domain to a port-channel egress interface

This task assigns a TVF domain to a port-channel egress interface.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface port-channel** command to access port-channel configuration mode.

```
device(config)# interface port-channel 10
```

3. Enter the **tvf-domain** command, specifying one of the valid formats.
   - To assign one TVF domain to the interface, specify its integer ID.

```
device(config-Port-channel-10)# tvf-domain add 10
```

   - To assign a range of TVF domains to the interface, insert a hyphen (-) between the beginning and ending integers.

```
device(config-Port-channel-10)# tvf-domain add 20-30
```

   - To assign individual domains and ranges of domains, separate them with commas (for example, 1,5-7,55).

```
device(config-Port-channel-10)# tvf-domain add 1,5-7,55
```

   - To assign all defined TVF domains to the interface, enter **tvf-domain all**.

```
device(config-Port-channel-10)# tvf-domain all
```

   - To assign all defined TVF domains to the interface—except for those specified—enter the **tvf-domain except** option.

```
device(config-Port-channel-10)# tvf-domain except 1,2,4-7
```

Replicating traffic to multiple interfaces

This task replicates traffic entering an interface to multiple egress interfaces.

1. Create needed TVF domains, as described in Creating TVF domains on page 28.

```
device# configure terminal
device(config)# tvf-domain 10
device(config-tvf-domain-10)# exit
```

2. Assign TVF domains to the egress interfaces.
   - Physical interfaces (For details, refer to Assigning a TVF domain to a physical egress interface on page 29.)

```
device(config)# interface ethernet 0/5
device(conf-if-eth-0/5)# tvf-domain add 10
```

   - Port-channel interfaces

```
device(config)# interface port-channel 10
device(config-Port-channel-10)# tvf-domain add 10
```

3. Configure a regular ACL or a user-defined ACL (UDA):
   - For a regular ACL, refer to Creating ACLs for NPB on page 18.

     ```
     device(config)# ip access-list standard aclNPB_01
     device(conf-ipacl-std)# permit host 192.1.1.1 count
     device(conf-ipacl-std)# exit
     ```

   - For a UDA, refer to UDAs on page 19.
4. Configure a route map that contains the relevant **match { mac | ip | ipv6 | uda } address acl** command.

   ```
   device(config)# route-map npb_map1 permit 1
   device(config-route-map-npb_map1/permit/1)# match ip address acl acl_2
   ```

5. Specify the TVF domain that contains the egress interfaces for the replicated traffic.
   - Without stripping 802.1q VLAN tags:

     ```
     device(config-route-map-npb_map1/permit/1)# set next-hop-tvf-domain 5
     ```

   - Stripping 802.1q VLAN tags:

     ```
     device(config-route-map-npb_map1/permit/1)# set next-hop-tvf-domain 5 strip-vlan outer
     ```

   - Adding a 802.1q VLAN tag (optionally, in addition to **strip-vlan outer**):

     ```
     device(config-route-map-npb_map1/permit/1)# set interface ethernet 0/5 add-vlan outer 2
     ```

6. Exit to global configuration mode.

   ```
   device(config-route-map-npb_map1/permit/1)# exit
   ```

7. Apply the route map to the ingress interface.
   - Physical interface

     ```
     device(config)# interface ethernet 0/2
     device(conf-if-eth-0/2)# npb policy route-map npb_map1
     ```

   - Port-channel interface

     ```
     device(config)# interface port-channel 10
     device(config-Port-channel-10)# npb policy route-map npb_map1
     ```

The following example replicates traffic entering an interface to multiple egress interfaces.

```
device# configure terminal
device(config)# tvf-domain 10
device(config-tvf-domain-10)# description Sample TVF description
device(config-tvf-domain-10)# exit
device(config)# interface ethernet 0/5
device(conf-if-eth-0/5)# tvf-domain add 10
device(conf-if-eth-0/5)# exit
device(config)# interface port-channel 10
device(config-Port-channel-10)# tvf-domain add 10
device(config-Port-channel-10)# exit
device(config)# route-map npb_map1 permit 1
device(config-route-map-npb_map1/permit/1)# match ip address acl acl_2
device(config-route-map-npb_map1/permit/1)# set next-hop-tvf-domain 5
device(config-route-map-npb_map1/permit/1)# exit
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# npb policy route-map npb_map1
```

Load Balancing Under NPB

Load balancing overview

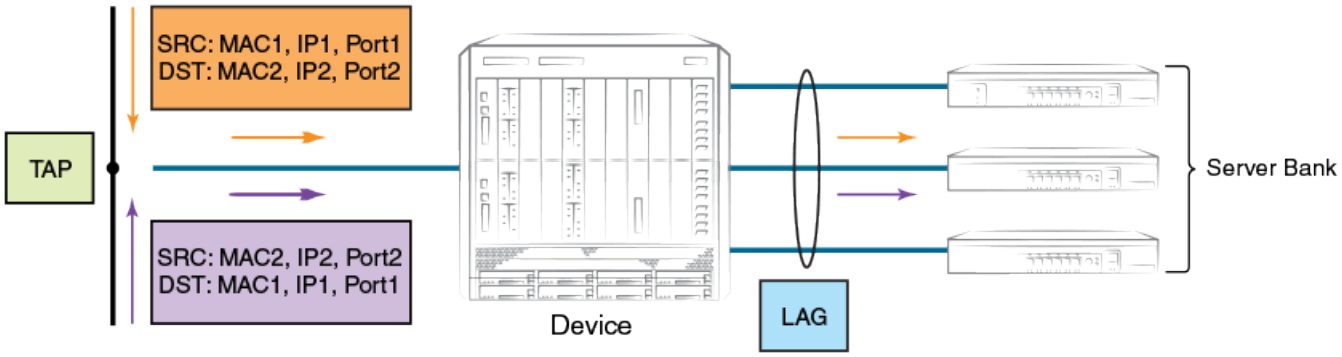
The only type of link-aggregation load-balancing supported under Network Packet Broker (NPB) is symmetric load balancing.

NOTE
For details on link aggregation, including link-aggregation groups (LAGs), refer to the "Link Aggregation" section of the *Extreme SLX-OS Layer 2 Switching Configuration Guide*. LAGs are also referred to as *port-channels*.

Symmetric load-balancing interchanges source and destination addresses to ensure that bidirectional traffic between a source and destination pair flows through one LAG member. Under NPB, symmetric load-balancing is enabled by default and cannot be disabled. However, you can modify the load-balancing parameters.

For network telemetry applications, network traffic is tapped and sent to a device that hashes selected traffic to the application servers downstream. For many monitoring and security applications, bidirectional conversations flowing through the system must be carried on the same port of a port-channel (LAG). In addition, firewalls between devices can be configured to allow such bidirectional conversations.

FIGURE 1 Symmetric load balancing



Load balancing of tunneled frames, with header stripping

Load balancing of tunneled frames (VXLAN, NVGRE, ERSPAN, GTP, or MPLS pseudo-wire) is affected by header stripping:

- Without header stripping, load balancing is based on the hash produced from outer headers.
- With header stripping, load balancing is based on the hash produced from inner headers.

NOTE
For details of header stripping, refer to Interface-level header stripping on page 40.

Configuring symmetric load balancing

Use this task to change the load-balancing mode from the default **src-dst-ip-mac-vid-port** mode or another current mode to the load-balancing mode that you require.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **load-balance** command to specify the mode that you require.

```
device(conf)# load-balance src-dst-ip
```

Symmetric load-balancing options and examples

The following tables display examples of the NPB symmetric load-balancing hashing options.

src-dst-ip

The following table displays inputs for a **load-balance src-dst-ip** example. The distribution is based on source and destination IPv4 or IPv6 addresses.

TABLE 24 Symmetric load-balancing **src-dst-ip** option

Flow	Source IP	Destination IP	Source MAC	Destination MAC	VLAN
1	IPA	IPB	MACA	MACB	—
2	IPB	IPA	MACD	MACE	—
3	IPD	IPE	MACA	MACB	VID2
4	IPA	IPC	MACA	MACD	VID2

Because flows 1 and 2 share the same source and destination IP addresses (the other fields are not considered), they are load balanced on the same port-channel port.

src-dst-ip-mac-vid

The following table displays inputs for a **load-balance src-dst-ip-mac-vid** example. The distribution is based on source and destination IPv4 or IPv6 and MAC addresses; and outer VLAN ID (VID).

TABLE 25 Symmetric load-balancing **src-dst-ip-mac-vid** option

Flow	Source IP	Destination IP	Source MAC	Destination MAC	VLAN
1	IPA	IPB	MACA	MACB	VID1
2	IPB	IPA	MACB	MACA	VID1
3	IPA	IPB	MACA	MACB	VID2

TABLE 25 Symmetric load-balancing **src-dst-ip-mac-vid** option (continued)

Flow	Source IP	Destination IP	Source MAC	Destination MAC	VLAN
4	IPA	IPB	MACD	MACE	VID2

Because flows 1 and 2 share the same source and destination IP and MAC addresses and the same VLAN, they are load-balanced on the same port-channel port.

src-dst-ip-mac-vid-port

The following table displays inputs for a **load-balance src-dst-ip-mac-vid-port** example. The distribution is based on source and destination IPv4 or IPv6 and MAC addresses, VID, and TCP or UDP destination port.

NOTE
This is the default **load-balance** option.

TABLE 26 Symmetric load-balancing **src-dst-ip-mac-vid-port** option

Flow	Source IP	Destination IP	Source MAC	Destination MAC	Source L4 Port	Destination L4 Port	VLAN
1	IPA	IPB	MACA	MACB	P1	P2	VID1
2	IPB	IPA	MACB	MACA	P2	P1	VID1
3	IPA	IPB	MACA	MACB	P2	P1	VID2
4	IPA	IPB	MACD	MACE	P3	P4	VID2

Because flows 1 and 2 share the same source and destination IP and MAC addresses, the same TCP or UDP ports, and the same VLAN, they are load-balanced on the same port-channel port.

src-dst-ip-port

The following table displays inputs for a **load-balance src-dst-ip-port** example. The distribution is based on source and destination IPv4 or IPv6 address and TCP or UDP destination port.

TABLE 27 Symmetric load-balancing **src-dst-ip-port** option

Flow	Source IP	Destination IP	Source L4 Port	Destination L4 Port
1	IPA	IPB	P1	P2
2	IPB	IPA	P2	P1
3	IPA	IPB	P2	P1
4	IPA	IPB	P3	P4

Because flows 1 and 2 share the same source and destination IP addresses and the same TCP or UDP ports, they are load-balanced on the same port-channel port.

src-dst-mac-vid

The following table displays inputs for a **load-balance src-dst-mac-vid** example. The distribution is based on the source and destination MAC addresses and VID.

TABLE 28 Symmetric load-balancing **src-dst-mac-vid** option

Flow	Source MAC	Destination MAC	VLAN
1	MACA	MACB	VID1
2	MACB	MACA	VID1
3	MACA	MACB	VID2
4	MACD	MACE	VID2

Because flows 1 and 2 share the same source and destination MAC addresses and the same VLAN, they are load-balanced on the same port-channel port.

Forwarding traffic to a port-channel

For load balancing, this task forwards traffic entering an interface to a port-channel.

1. Configure a regular ACL or a user-defined ACL (UDA):
   - For a regular ACL, refer to Creating ACLs for NPB on page 18.

     ```
     device(config)# ip access-list standard aclNPB_01
     device(conf-ipacl-std)# permit host 192.1.1.1 count
     device(conf-ipacl-std)# exit
     ```

   - For a UDA, refer to UDAs on page 19.
2. Configure a route map that contains the relevant **match { mac | ip | ipv6 | uda } address acl** command.

   ```
   device(config)# route-map npb_map1 permit 1
   device(config-route-map-npb_map1/permit/1)# match ip address acl aclNPB_01
   ```

3. Specify the egress port-channel interface.
   - Without stripping 802.1q VLAN tags:

     ```
     device(config-route-map-npb_map1/permit/1)# set interface port-channel 10
     ```

   - Stripping 802.1q VLAN tags:

     ```
     device(config-route-map-npb_map1/permit/1)# set interface port-channel 10 strip-vlan outer
     ```

   - Adding a 802.1q VLAN tag (optionally, in addition to **strip-vlan outer**):

     ```
     device(config-route-map-npb_map1/permit/1)# set interface ethernet 0/5 add-vlan outer 2
     ```

4. Exit to global configuration mode.

```
device(config-route-map-npb_map1/permit/1)# exit
```

5. Apply the route map to a single ingress interface:
   - Physical interface

```
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# npb policy route-map npb_map1
```

   - Port-channel interface

```
device(config)# port-channel 5
device(config-Port-channel-10)# npb policy route-map npb_map1
```

The following example forwards traffic entering a physical interface to a port-channel.

```
device# configure terminal
device(config)# ip access-list standard aclNPB_01
device(conf-ipacl-std)# permit host 192.1.1.1 count
device(conf-ipacl-std)# exit
device(config)# route-map npb_map1 permit 1
device(config-route-map-npb_map1/permit/1)# match ip address acl aclNPB_01
device(config-route-map-npb_map1/permit/1)# set interface port-channel 10
device(config-route-map-npb_map1/permit/1)# exit
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# npb policy route-map npb_map1
```

Header Modification

Header-modification overview

Protocol headers help packets reach their destinations, but are not needed by the security and monitoring tools to which NPB forwards traffic.

Tagging and encapsulation techniques have long been a part of networking. However, the recent adoption of new encapsulation protocols—such as VXLAN, VN-Tag, and 802.1BR—can create visibility blind spots, because some visibility applications were not designed to interpret these new protocols.

By removing the encapsulation header, the NPB removes the burden of interpreting the various encapsulation protocols from the visibility applications. Therefore, the header-stripping feature enables network operators to deploy new encapsulation protocols without interfering with proper functioning of previously deployed visibility applications. Other header stripping benefits include:

- Reduced packet overhead and better visibility-application bandwidth utilization.
- Application of defined Layer 2 and Layer 3 ACLs to the inner headers.
- Load balancing based on the inner headers.

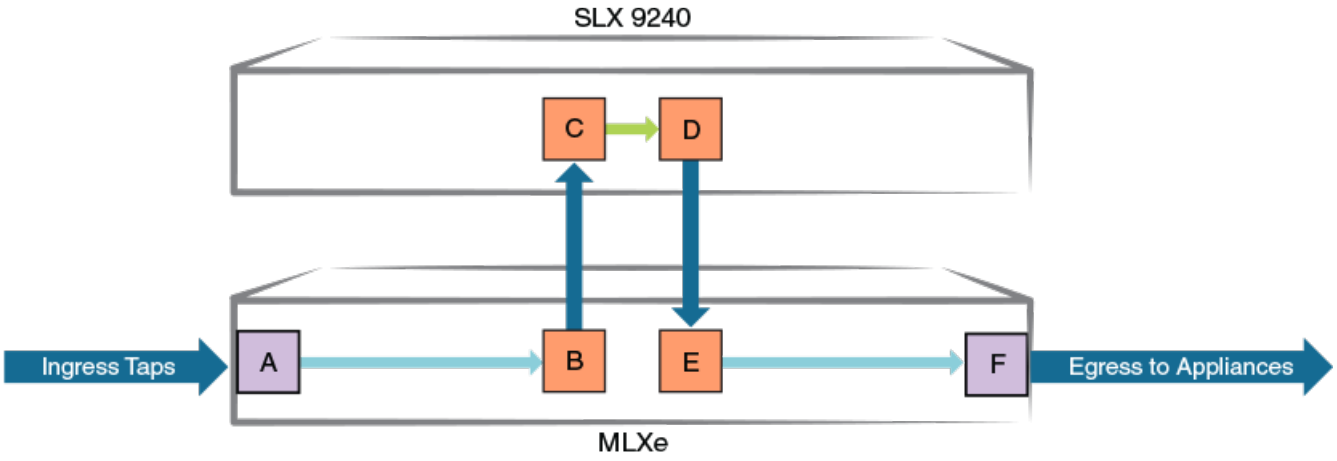
The following types of header modification are supported:

- Interface-level header stripping on page 40
- Configuring GTP-HTTPS frame filtering on page 50
- VLAN-header flow modification on page 51

Header-modification flow

The following figure displays a sample header-modification flow.

FIGURE 2 Header-modification flow



In the example flow, headers or frames of a specified type are removed, as follows:

1. Traffic enters an MLXe through port A.
2. The traffic exits the MLXe from port B and enters the SLX 9240 through port C. If a header-modification command is enabled on port C, specified headers or frames are removed from the flow.
3. Traffic is forwarded from SLX 9240 port C to port D.
4. Traffic is forwarded back to MLXe port E, using an NPB route-map. (For details, refer to Route maps under NPB on page 14.)
5. Traffic is forwarded from MLXe port E to port F, using an NPB route-map.
6. Traffic is forwarded from MLXe port F to monitoring tools.

Interface-level header stripping

Interface-level header stripping enables you to configure different types of header stripping on different interfaces.

Header-stripping configuration guidelines

You cannot configure 802.1BR and VN-Tag header-stripping on the same interface.

You can configure other combinations of header-stripping types on an interface. However, only one kind of stripping operation is actually implemented. For more detailed guidelines, refer to the "Configuration guidelines" sections for the various header types:

- 802.1BR header stripping on page 41
- VN-Tag header stripping on page 42
- VXLAN header stripping on page 44
- NVGRE header stripping on page 45

- ERSPAN-II header-stripping on page 46
- MPLS header stripping on page 47
- GTP de-encapsulation on page 49

Header-stripping configuration guidelines (tunneled frames)

The following header-stripping configuration guidelines apply to tunneled frames (VXLAN, NVGRE, ERSPAN, GTP, or MPLS pseudo-wire):

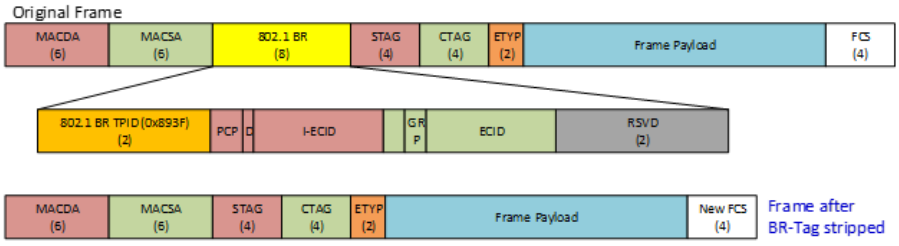
- Header stripping affects only the outer, encapsulation headers.
- Without header stripping, Layer 2 and Layer 3 ACLs act on outer headers; with header stripping, ACLs act on the inner headers.
- For load balancing configuration guidelines (including for tunneled frames with header stripping), refer to Load balancing of tunneled frames, with header stripping on page 33.

802.1BR header stripping

The following figure shows frame structure before and after 802.1BR header stripping.

Stripping the 802.1BR header prepares the header and frame payload for forwarding and processing.

FIGURE 3 Before and after 802.1BR header stripping



Configuration guidelines for 802.1BR header stripping

By default, interfaces support 802.1BR tags but not VN-Tags. For instructions how to toggle between the two header modes, refer to Configuring 802.1BR header stripping on page 42 and Configuring VN-Tag header stripping on page 43.

If a tunneled frame has an 802.1BR tag in the outer L2 header, VXLAN, NVGRE, ERSPAN, or L2-over-MPLS/pseudo-wire header-stripping also deletes the 802.1BR tag. (802.1BR tags in the inner L2 header are not supported.)

If both MPLS and 802.1BR header-stripping are configured, MPLS is always implemented. However, details vary between IP-over-MPLS and L2-over-MPLS regarding 802.1BR tags:

- Under IP-over-MPLS, only the MPLS labels are stripped; 802.1BR tags are not affected. The header diagram for this case is as follows:

IP-over-MPLS frame headers			
L2 (802.1BR tag)	MPLS	IPv4	TCP

- Under L2-over-MPLS (pseudo-wire), both the outer L2 (with 802.1BR tag) and the MPLS header are stripped. The header diagram for this case is as follows:

L2-over-MPLS frame (pseudo-wire) headers				
L2 (802.1BR tag)	MPLS	L2	IPv4	TCP

Configuring 802.1BR header stripping

Use this task to enable or disable 802.1BR header stripping on an Ethernet interface.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to access Ethernet interface configuration mode.

```
device(config)# interface ethernet 0/2
```

3. If default support for 802.1BR headers was changed on that interface to support for VN-Tag headers, enter **no allow-vn-tag** to restore support for 802.1BR headers.

```
device(conf-if-eth-0/2)# no allow-vn-tag
```

4. Enable or disable this feature on the interface.
   - To enable 802.1BR header stripping, enter **strip-802-1br**.

```
device(conf-if-eth-0/2)# strip-802-1br
```

   - To disable 802.1BR header stripping, enter **no strip-802-1br**.

```
device(conf-if-eth-0/2)# no strip-802-1br
```

The following example enables 802.1BR header stripping on an interface.

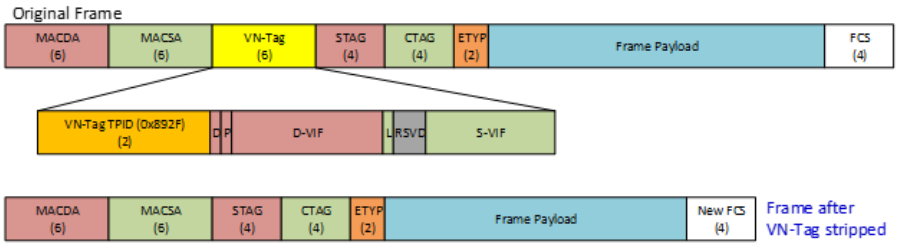
```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-/2)# strip-802-1br
```

VN-Tag header stripping

The following figure shows frame structure before and after 802.1BR header stripping.

Stripping the VN-Tag prepares the header and frame payload for forwarding and processing.

FIGURE 4 Before and after VN-Tag header stripping



Configuration guidelines for VN-Tag header stripping

By default, interfaces support 802.1BR tags but not VN-Tags. For instructions how to toggle between the two header modes, refer to Configuring 802.1BR header stripping on page 42 and Configuring VN-Tag header stripping on page 43.

If a tunneled frame has a VN-Tag in the outer L2 header, VXLAN, NVGRE, ERSPAN, or L2-over-MPLS/pseudo-wire header-stripping also deletes the VN-Tag. (VN-Tags in the inner L2 header are not supported.)

If both MPLS and VN-Tag header-stripping are configured, MPLS is always implemented. However, details vary between IP-over-MPLS and L2-over-MPLS regarding VN-Tags:

- Under IP-over-MPLS, only the MPLS labels are stripped; VN-Tags are not affected. The header diagram for this case is as follows:

IP-over-MPLS frame headers			
L2 (VN-Tag)	MPLS	IPv4	TCP

- Under L2-over-MPLS (pseudo-wire), both the outer L2 (with VN-Tag) and the MPLS header are stripped. The header diagram for this case is as follows:

IP-over-MPLS frame (pseudo-wire) headers				
L2 (VN-Tag)	MPLS	L2	IPv4	TCP

Configuring VN-Tag header stripping

Use this task to enable or disable Virtual NIC (VN)-Tag header stripping on an Ethernet interface.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to access Ethernet interface configuration mode.

```
device(config)# interface ethernet 0/2
```

3. If default support for 802.1BR headers is in effect on that interface, enter **allow-vn-tag** to support for VN-Tag headers.

```
device(conf-if-eth-0/2)# allow-vn-tag
```

4. Enable or disable this feature on the interface.
   - To enable VN-Tag header stripping, enter **strip-vn-tag**.

```
device(conf-if-eth-0/2)# strip-vn-tag
```

   - To disable VN-Tag header stripping, enter **no strip-vn-tag**.

```
device(conf-if-eth-0/2)# no strip-vn-tag
```

The following example enables VN-Tag header stripping on an interface.

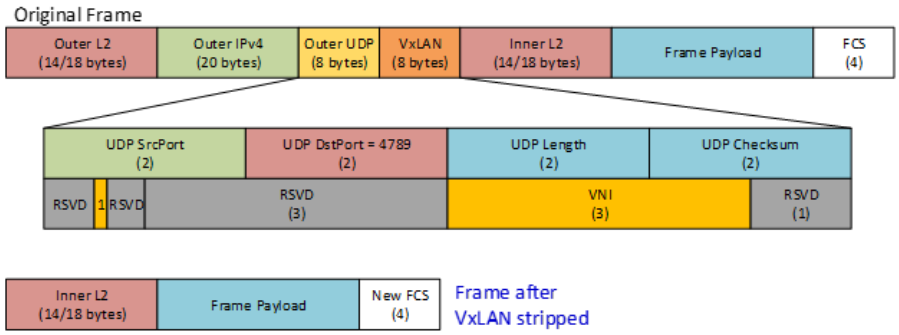
```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# strip-vn-tag
```

VXLAN header stripping

The following figure shows frame structure before and after VXLAN header-stripping.

VXLAN-encapsulated frames have an outer L2-IPv4-UDP-VXLAN header structure. Stripping the outer headers prepares the inner headers and frame payload for forwarding and processing.

FIGURE 5 Before and after VXLAN header stripping



Configuration guidelines for VXLAN header stripping

For tunneled frames, header-stripping affects only the outer, encapsulation headers. For example:

- If both ERSPAN and VXLAN header-stripping are configured on the ingress interface, only the ERSPAN headers are stripped. The header diagram for this case is as follows:

ERSPAN headers				VXLAN headers				Payload frame headers		
L2	IPv4	GRE	ERSPAN	L2	iIPv4	UPD	VXLAN	L2	IPv4	TCP

- If both VXLAN and MPLS header-stripping are configured, only VXLAN headers are stripped. The header diagram for this case is as follows:

VXLAN headers				Payload frame headers			
L2	IPv4	UDP	VXLAN	L2	MPLS	IPv4	TCP

Configuring VXLAN header stripping

Use this task to enable or disable Virtual Extensible LAN (VXLAN) header stripping on an Ethernet interface.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to access Ethernet interface configuration mode.

```
device(config)# interface ethernet 0/2
```

3. Enable or disable this feature on the interface.
   - To enable VXLAN header stripping, enter **strip-vxlan**.

     ```
     device(conf-if-eth-0/2)# strip-vxlan
     ```

   - To disable VXLAN header stripping, enter **no strip-vxlan**.

     ```
     device(conf-if-eth-0/2)# no strip-vxlan
     ```

The following example enables VXLAN header stripping on an interface.

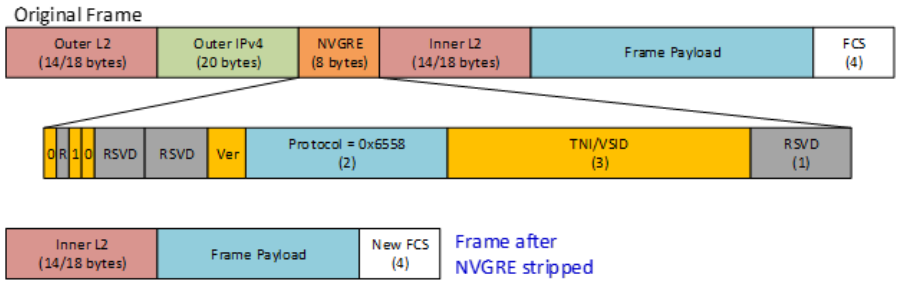
```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# strip-vxlan
```

NVGRE header stripping

The following figure shows frame structure before and after NVGRE header stripping.

NVGRE-encapsulated frames have an outer L2-IPv4-NVGRE header structure. Stripping the outer headers prepares the inner headers and frame payload for forwarding and processing.

FIGURE 6 Before and after NVGRE header stripping



Configuration guidelines for NVGRE header stripping

For tunneled frames, header-stripping affects only the outer, encapsulation headers. For example:

- If both ERSPAN and NVGRE header-stripping are configured on the ingress interface, only the ERSPAN headers are stripped. (NVGRE header-stripping alone has no effect.) The header diagram for this case is as follows:

ERSPAN headers				NVGRE headers			Payload frame headers		
L2	IPv4	GRE	ERSPAN	L2	IPv4	NVGRE	L2	IPv4	TCP

- If both NVGRE and MPLS header-stripping are configured, only NVGRE headers are stripped; MPLS labels are left untouched. The header diagram for this case is as follows:

NVGRE headers			Payload frame headers			
L2	IPv4	NVGRE	L2	MPLS	IPv4	TCP

Configuring NVGRE header stripping

Use this task to enable or disable Network Virtualization using Generic Routing Encapsulation (NVGRE) header stripping on an Ethernet interface.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to access Ethernet interface configuration mode.

```
device(config)# interface ethernet 0/2
```

3. Enable or disable this feature on the interface.
   - To enable NVGRE header stripping, enter **strip-nvgre**.

```
device(conf-if-eth-0/2)# strip-nvgre
```

   - To disable NVGRE header stripping, enter **no strip-nvgre**.

```
device(conf-if-eth-0/2)# no strip-nvgre
```

The following example enables NVGRE header stripping on an interface.

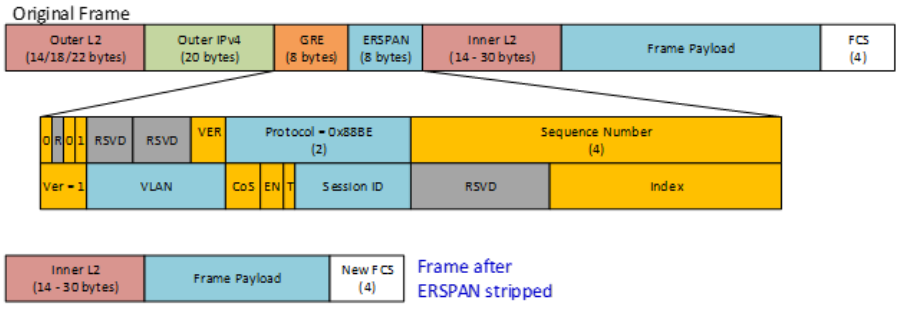
```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# strip-nvgre
```

ERSPAN-II header-stripping

The following figure shows frame structure before and after ERSPAN Type II header stripping.

ERSPAN-II-encapsulated frames have an outer L2-IPv4-GRE-ERSPAN header structure. Stripping the outer headers prepares the inner headers and frame payload for forwarding and processing.

FIGURE 7 Before and after ERSPAN-II header-stripping



Configuration guidelines for ERSPAN-II header-stripping

For tunneled frames, header-stripping affects only the outer, encapsulation headers. For example:

- If both ERSPAN and VXLAN header-stripping are configured on the ingress interface, only the ERSPAN headers are stripped. (VXLAN header-stripping alone has no effect.) The header diagram for this case is as follows:

ERSPAN headers				VXLAN headers				Payload frame headers		
L2	IPv4	GRE	ERSPAN	L2	IPv4	GRE	ERSPAN	L2	IPv4	TCP

- If both ERSPAN and MPLS header-stripping are configured, only ERSPAN headers are stripped; MPLS labels are left untouched. The header diagram for this case is as follows:

ERSPAN headers				Payload frame headers			
L2	IPv4	GRE	ERSPAN	L2	MPLS	IPv4	TCP

Configuring ERSPAN-II header stripping

Use this task to enable or disable Encapsulated Remote Switch Port Analyzer (ERSPAN-II) header stripping on an Ethernet interface.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to access Ethernet interface configuration mode.

```
device(config)# interface ethernet 0/2
```

3. Enable or disable this feature on the interface.
   - To enable ERSPAN-II header stripping, enter **strip-erspan**.

```
device(conf-if-eth-0/2)# strip-erspan
```

   - To disable ERSPAN-II header stripping, enter **no strip-erspan**.

```
device(conf-if-eth-0/2)# no strip-erspan
```

The following example enables ERSPAN-II header stripping on an interface.

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

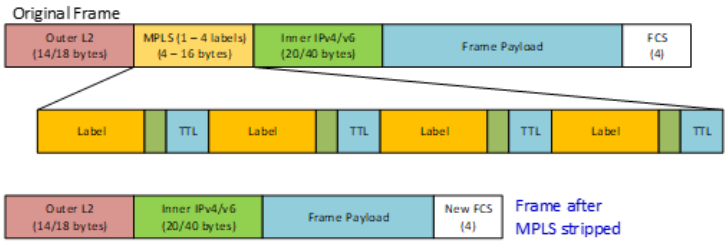
MPLS header stripping

MPLS header stripping is effective for both IP payload and pseudo-wire.

IP payload

The following diagram shows frame structure before and after MPLS header stripping for an encapsulated IP payload. Stripping the MPLS headers prepares the inner headers and frame payload for forwarding and processing.

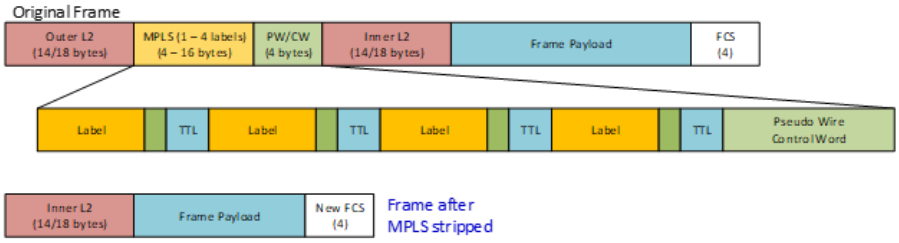
FIGURE 8 Before and after MPLS header stripping (IP payload)



Pseudo-wire

The following diagram shows frame structure before and after MPLS header stripping for an entire encapsulated Ethernet frame (pseudo-wire). Stripping the outer headers (MPLS labels, outer L2, and the pseudo-wire control word) prepares the inner headers and frame payload for forwarding and processing.

FIGURE 9 Before and after MPLS header stripping (pseudo-wire)



Configuration guidelines for MPLS header stripping

For tunneled frames, header-stripping affects only the outer, encapsulation headers. For example, if both VXLAN and MPLS header-stripping are configured, only VXLAN headers are stripped. The header diagram for this case is as follows:

VXLAN headers				Payload frame headers			
L2	IPv4	UDP	VXLAN	L2	MPLS	IPv4	TCP

If both MPLS and 802.1BR or VN-Tag header-stripping are configured, MPLS is always implemented. However, details vary between IP-over-MPLS and L2-over-MPLS regarding 802.1BR or VN-Tags:

- Under IP-over-MPLS, only the MPLS labels are stripped; 802.1BR or VN-Tags are not affected. The header diagram for this case is as follows:

IP-over-MPLS frame headers			
L2 (802.1BR or VN-Tag)	MPLS	IPv4	TCP

- Under L2-over-MPLS (pseudo-wire), both the outer L2 (with 802.1BR or VN-Tag) and the MPLS header are stripped. The header diagram for this case is as follows:

IP-over-MPLS frame (pseudo-wire) headers				
L2 (802.1BR or VN-Tag)	MPLS	L2	IPv4	TCP

Configuring MPLS header stripping

Use this task to enable or disable Multi-Protocol Label Switching (MPLS) header stripping on an Ethernet interface.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to access Ethernet interface configuration mode.

```
device(config)# interface ethernet 0/2
```

3. Enable or disable this feature on the interface.
   - To enable MPLS header stripping, enter **strip-mpls**.

```
device(conf-if-eth-0/2)# strip-mpls
```

   - To disable MPLS header stripping, enter **no strip-mpls**.

```
device(conf-if-eth-0/2)# no strip-mpls
```

The following example enables MPLS header stripping on an interface.

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

GTP de-encapsulation

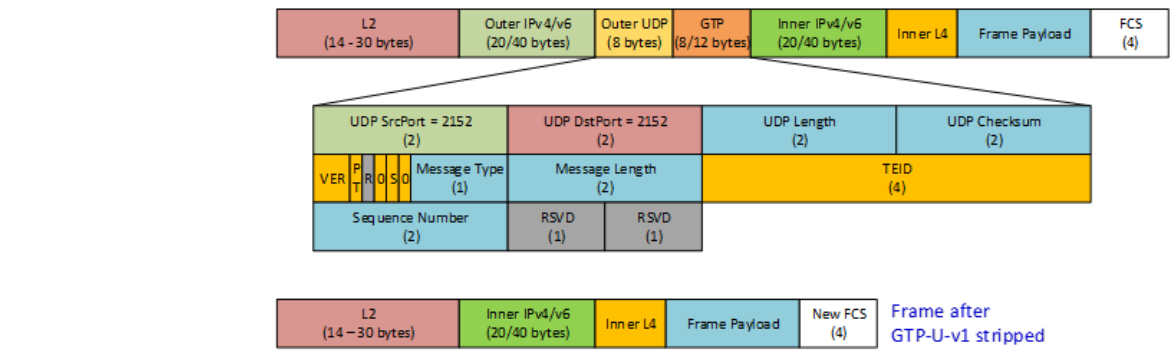
The following figure shows frame structure before and after GTP de-encapsulation.

GTP-U-v1-encapsulated frames have an inner structure of IPv4/v6 tunneled in L2-IPv4/v6-UDP-GTP headers. When GTP de-encapsulation (header-stripping) is enabled, the outer IP, UDP, and GTP headers are discarded, but the outer L2 header is retained.

Following such de-encapsulation, ACLs are applied as follows:

- Layer 2 ACLs apply to the outer L2 header.
- Layer 3 ACLs apply to the inner IP header.

FIGURE 10 Before and after GTP de-encapsulation



GTP de-encapsulation configuration guidelines

GTP de-encapsulation is supported only for GTP v.1 frames, with or without a sequence number.

When GTP de-encapsulation is performed on a frame, only one C-tag is retained in the L2 header. Other tags—802.1BR, VN-Tag, S-Tag, and Outer C-Tag—are dropped from the L2 header.

If GTP de-encapsulation is applied to a frame, VLAN-header modification settings are ignored on that frame. For details, refer to VLAN-header flow modification on page 51.

Configuring GTP de-encapsulation

Use this task to enable or disable GTP de-encapsulation on an Ethernet interface.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Enter the **interface ethernet** command to access Ethernet interface configuration mode.

```
device(config)# interface ethernet 0/2
```

3. Enable or disable this feature on the interface.
   - To enable GTP de-encapsulation, enter **gtp-de-encapsulation**.

```
device(conf-if-eth-0/2)# gtp-de-encapsulation
```

   - To disable GTP de-encapsulation, enter **no strip-vxlan**.

```
device(conf-if-eth-0/2)# no gtp-de-encapsulation
```

The following example enables GTP de-encapsulation on an interface.

```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# gtp-de-encapsulation
```

Configuring GTP-HTTPS frame filtering

Use this task to enable and disable dropping GPRS Tunneling Protocol (GTP)-v.1 frames that encapsulate HTTPS packets.

NOTE
For dropped GTP-HTTPS frames, Interface-level header stripping on page 40 is not relevant.

1. Enter **configure terminal** to access global configuration mode.

```
device# configure terminal
```

2. Access a physical interface on which you need to configure this feature.

```
device(conf)# interface ethernet 0/5
```

3. Enable or disable this feature on the interface.
   - To enable dropping GTP frames that encapsulate HTTPS packets, enter **deny inner-gtp-https**.

     ```
     device(conf-if-eth-0/5)# deny inner-gtp-https
     ```

   - To disable dropping GTP frames that encapsulate HTTPS packets, enter **no deny inner-gtp-https**.

     ```
     device(conf-if-eth-0/5)#  no deny inner-gtp-https
     ```

VLAN-header flow modification

NPB supports 802.1q VLAN outer-header stripping and addition for aggregation, replication, and load balancing, as described in the following table. For implementation details, refer to the linked tasks.

VLAN-header modification is implemented at flow level. The other types of header modification under NPB are implemented at interface level:

- Interface-level header stripping on page 40
- Configuring GTP-HTTPS frame filtering on page 50

In general, interface-level header-stripping and VLAN-header modification can apply concurrently. The only exception is GTP de-encapsulation: If GTP de-encapsulation is applied to a frame, VLAN-header modification settings are ignored on that frame.

TABLE 29 VLAN-header modification commands and tasks

Feature	**set** command	Task
Aggregation	**set interface**	Aggregating traffic from interfaces on page 27
Replication	**set next-hop-tvf-domain**	Replicating traffic to multiple interfaces on page 30
Load balancing	**set interface**	Forwarding traffic to a port-channel on page 36

Appendix A: NPB Command Reference

allow-vn-tag

Changes an interface from the default support for 802.1BR headers to support for VN-Tag headers.

Syntax

allow-vn-tag

no allow-vn-tag

Command Default

802.1BR headers are supported and VN-Tag headers are not supported.

Modes

Interface configuration mode

Usage Guidelines

To learn where default support for 802.1BR headers was changed to support for VN-Tag headers—by running **allow-vn-tag**—run the **show running-config interface** command:

- VN-Tag interfaces display `allow-vn-tag`.
- 802.1BR interfaces do not display `allow-vn-tag`.

This command is supported only under network packet-broker (NPB) system-mode.

Disabling 802.1BR header-mode on an interface also disables E-Tag support on that interface.

After you run this command, you do not need to reboot SLX-OS.

The **no** form of this command disables support for VN-Tag headers and restores the default support for 802.1BR headers.

Examples

The following example enables VN-Tag header-mode on an interface, disabling the default 802.1BR mode.

```
device# configure terminal
device(config)# interface Ethernet 0/1
device(config-if-eth-0/1)# allow-vn-tag
```

The following example restores the default 802.1BR header-mode on an interface, disabling support for VN-Tag headers.

```
device# configure terminal
device(config)# interface Ethernet 0/1
device(config-if-eth-0/1)# no allow-vn-tag
```

History

Release version	Command history
18s.1.00	This command was introduced.

clear counters access-list

For a given network protocol and inbound/outbound direction, clears ACL statistical information. You can clear all statistics for a specified ACL or only for that ACL on a specified interface. You can also clear statistical information for all ACLs bound to a specified Ethernet interface, VLAN, or VE.

Syntax

clear counters access-list interface { **ethernet** *0* **/** *port* | **port-channel** *index* | **vlan** *vlan_id* } { **in** | **out** }

clear counters access-list interface ve *vlan_id* { **in** | **out** }

clear counters access-list { **ip** | **ipv6** | **mac** } [*acl-name* { **in** | **out** }]

clear counters access-list { **ip** | **ipv6** } *acl-name* **interface** { **ethernet** *slot* **/** *port* | **port-channel** *index* | **ve** *vlan_id* } { **in** | **out** }

clear counters access-list { **ip** | **ipv6** } [*acl-name* { **global in** }]

clear counters access-list mac *acl-name* **interface** { **ethernet** *slot* **/** *port* | **port-channel** *index* | **vlan** *vlan_id* } { **in** | **out** }

Parameters

interface
Specifies an interface.

ethernet
Specifies a physical Ethernet interface.

0
Specifies a valid slot number. The only valid slot number is 0.

port
Specifies a valid port number.

port-channel *number*
Specifies a port-channel. Available channels range from 1 through 6144.

in | **out**
Specifies the binding direction (incoming or outgoing).

vlan *vlan_id*
(Available only on Layer 2) Specifies a VLAN.

ve *vlan_id*
(Available only on Layer 3) Specifies a virtual Ethernet (VE) interface.

ip | **ipv6** | **mac**
Specifies the network protocol.

global
Specifies Level 3 receive ACLs (rACLs), which are applied at device-level, rather than at interface-level.

mac *acl-name*
Specifies the MAC ACL name. To clear statistics on all counters of an ACL-type, do not specify *acl-name*.

in | **out**
Specifies the binding direction (incoming or outgoing).

Modes

Privileged EXEC mode

Examples

The following example clears ACL statistics on a specified Ethernet interface.

```
device# clear counters access-list interface ethernet 0/1
```

The following example clears ACL statistics for a specified MAC ACL on a specified Ethernet interface.

```
device# clear counters access-list mac MAC_ACL_1 interface ethernet 0/2
```

The following example clears ACL statistics for a specified MAC ACL on all interfaces on which this ACL is applied.

```
device# clear counters access-list mac MAC_ACL_1
```

The following example clears ACL statistics for a specified IPv4 ACL on a specified interface.

```
device# clear counters access-list ip IP_ACL_1 interface ethernet 0/3
```

The following example clears ACL statistics for a specified IPv4 ACL on all interfaces on which it is applied.

```
device# clear counters access-list ip IP_ACL_1
```

The following example clears incoming ACL statistics for a specified IPv6 ACL on a virtual Ethernet (VE) interface.

```
device# clear counters access-list ipv6 ip_acl_3 interface ve 10 in
```

The following example clears receive-path ACL statistics for a specified IPv6 ACL.

```
device# clear counters access-list ipv6 ipv6_acl_10 global in
```

History

Release version	Command history
17s.1.00	This command was introduced.

deny inner-gtp-https

Enables the dropping—from ingress traffic—of GPRS Tunneling Protocol (GTP) frames that encapsulate HTTPS packets.

Syntax

deny inner-gtp-https

no deny inner-gtp-https

Command Default

Such filtering is not enabled.

Modes

Interface configuration mode

Usage Guidelines

This feature must be implemented per physical interface.

This feature applies to GTP v.1 frames.

If this feature filtering is enabled, common practice is to forward the filtered traffic by using a Network Packet Broker (NPB) route-map.

The **no** form of this command disables this feature.

Examples

The following example enables dropping GTP frames that encapsulate HTTPS packets.

```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# deny inner-gtp-https
```

The following example restores the default setting of not dropping HTTPS packets transported within GTP frames.

```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# no deny inner-gtp-https
```

History

Release version	Command history
17s.1.02	This command was introduced.

description (TVF domain)

Describes a Transparent VLAN flooding (TVF) domain.

Syntax

description *description-text*

description

Command Default

TVF domains have no description.

Parameters

description-text

Describes a TVF domain.

Modes

TVF-domain configuration mode

Usage Guidelines

To remove a description, enter the command with no *description-text*.

To modify a description, enter the full **description** *description-text* command. You do not need to first remove the previous description.

Examples

The following example creates a TVF domain and then adds a description.

```
device# configure terminal
device(config)# tvf-domain 10
device(config-tvf-domain-10)# description Sample TVF domain
```

The following example displays current TVF domains and their descriptions, accesses a TVF domain, changes its description, and then displays the modifed description.

```
device# show running-config tvf-domain
tvf-domain 1
!
tvf-domain 10
description Sample TVF domain
!

device# configure terminal
device(config)# tvf-domain 10
device(config-tvf-domain-10)# description Enterprise TVF domain

device(config-tvf-domain-10)# do show running-config tvf-domain
tvf-domain 1
!
tvf-domain 10
description Enterprise TVF domain
```

History

Release version	Command history
18s.1.00	This command was introduced.

flow

In a UDA profile, specifies the expected packet structure.

Syntax

flow header0 *header0-type* [**header1** *header1-type* [**header2** *header2-type* [**header3** *header3-type* [**header4** *header4-type* [**header5** *header5-type* [**header6** *header6-type* [**header7** *header7-type*]]]]]]]

no flow header0 *header0-type* [**header1** *header1-type* [**header2** *header2-type* [**header3** *header3-type* [**header4** *header4-type* [**header5** *header5-type* [**header6** *header6-type* [**header7** *header7-type*]]]]]]]

Command Default

In a UDA profile, no expected packet-header types are defined.

Parameters

header0,1,2,3,4,5,6,7 *header0,1,2,3,4,5,6,7-type*

Specifies expected packet header-types. For supported values, refer to the Usage Guidelines.

Modes

UDA-profile configuration mode

Usage Guidelines

Ths command is supported only in NPB system mode.

You can specify up to eight headers types.

TABLE 30 Header types supported in UDA profiles

Value	Description
ETHERNET	Ethernet header
TUN_ETHERNET	Inner Ethernet header in tunneled frames
IPV4	IPv4 header
IPV6	IPv6 header
ARP	ARP header
TCP	TCP header
UDP	UDP header
SCTP	SCTP header
IGMP	IGMP header
ICMP	ICMP header
ICMPV6	ICMPv6 header
MPLS	MPLS header
GRE	GRE header

TABLE 30 Header types supported in UDA profiles (continued)

Value	Description
VXLAN	VXLAN header
GTP	GTP header
PAYLOAD4	Payload (4 bytes)
PAYLOAD8	Payload (8 bytes)
PAYLOAD16	Payload (16 bytes)
PAYLOAD32	Payload (32 bytes)

To delete a flow, use the **no** form of this command.

To modify a flow, delete the current flow and define a new one.

Examples

The following example configures a UDA profile, creates a flow, and specifies header types and fields.

```
device# configure terminal
device(config)# uda-key profile fkp1
device(conf-uda-profile-fkp1)# flow header0 ETHERNET header1 IPV4 header2 UDP header3 PAYLOAD32
device(conf-uda-profile-fkp1)# uda-key0 header0 ETHER_TYPE
device(conf-uda-profile-fkp1)# uda-key1 header1 PROTOCOL
device(conf-uda-profile-fkp1)# uda-key2 header2 DST_PORT
device(conf-uda-profile-fkp1)# uda-key3 header3 WORD1
```

History

Release version	Command history
18s.1.00	This command was introduced.

gtp-de-encapsulation

Enables the GPRS Tunneling Protocol (GTP) de-encapsulation feature on an interface.

Syntax

gtp-de-encapsulation

no gtp-de-encapsulation

Command Default

GTP de-encapsulation is not configured.

Modes

Physical interface configuration mode

Usage Guidelines

This feature is relevant only under NPB system mode, and supports only the GTPv1-U protocol.

This feature applies to both IPv4 and IPv6 traffic.

When GTP de-encapsulation is performed on a frame, only one C-tag is retained in the L2 header. Other tags—802.1BR, VN-Tag, S-Tag, and Outer C-Tag—are dropped from the L2 header.

If GTP de-encapsulation is applied to a frame, VLAN-header modification settings are ignored on that frame.

The final Frame Check Sequence (FCS) is updated with a recalculated CRC.

The **no** form of this command disables GTP de-encapsulation on the interface.

Examples

The following example enables the GTP de-encapsulation feature on an interface.

```
device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# gtp-de-encapsulation
```

The following example disables the GTP de-encapsulation feature on an interface.

```
device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# no gtp-de-encapsulation
```

History

Release version	Command history
18s.1.00	This command was introduced.

load-balance (NPB mode)

Configures the LAG load balancing settings, in Network Packet Broker (NPB) mode.

Syntax

load-balance { **dst-mac-vid** | **src-mac-vid** | **src-dst-mac-vid** | **src-dst-ip** | **src-dst-ip-mac-vid** | **src-dst-ip-port** | **src-dst-ip-mac-vid-port** }

no load-balance

Command Default

The default setting is **src-dst-ip-mac-vid-port**.

Parameters

dst-mac-vid
(Not supported in NPB mode) Specifies that the distribution is based on the destination MAC address and outer VLAN ID (VID).

src-mac-vid
(Not supported in NPB mode) Specifies that the distribution is based on the source MAC address and VID.

src-dst-mac-vid
Specifies that the distribution is based on the source and destination MAC addresses; and VID.

src-dst-ip
Specifies that the distribution is based on the source and destination IPv4 or IPv6 address.

src-dst-ip-mac-vid
Specifies that the distribution is based on the source and destination IPv4 or IPv6 addresses, MAC address, and VID.

src-dst-ip-port
Specifies that the distribution is based on the source and destination IPv4 or IPv6 addresses and TCP or UDP destination port.

src-dst-ip-mac-vid-port
Specifies that the distribution is based on the source and destination IPv4 or IPv6, MAC address, VID, and TCP or UDP destination port.

Modes

Global configuration mode

Usage Guidelines

The only type of load balancing supported in NPB mode is symmetric load-balance hashing.

In NPB mode, the VID component of keywords is ignored.

Use the **no** form of this command to return to the default setting.

Examples

The following example specifies that the distribution is based on source and destination MAC addresses and VLAN ID.

```
device# configure terminal
device(config)# load-balance src-dst-mac-vid
```

History

Release version	Command history
17s.1.00	The default system mode version of this command was introduced.
17s.1.02	The Network Packet Broker (NPB) version of this command was introduced.

match ip address acl (NPB)

In a route-map stanza, matches IPv4 address conditions specified in an IPv4 ACL.

Syntax

match ip address acl *acl-name*

no match ip address acl *acl-name*

Parameters

acl-name

Specifies an IPv4 ACL name unique among all ACLs (Layer 2 and Layer 3). The name can from 1 through 63 characters in length and must begin with an alphanumeric character. No special characters are allowed, except for the underscore and hyphen.

Modes

Route-map configuration mode

Usage Guidelines

A route-map stanza can contain only one **match { ip | ipv6 | mac | uda } address acl** statement.

The absence of a **match** statement is treated as "match any"; all traffic is forwarded according to the **set** statement.

Use the **no** form of this command to remove the match.

Examples

The following example creates an IPv4 ACL that permits traffic from a specific source IP and then includes that ACL in a route-map stanza.

```
device# configure terminal
device(config)# ip access-list standard aclNPB_01
device(conf-ipacl-std)# permit host 192.1.1.1 count
device(conf-ipacl-std)# exit
device(config)# route-map example1 permit 1
device(config-route-map-example1/permit/1)# match ip address acl acl_2
```

History

Release version	Command history
17s.1.01	This command was introduced.

match ipv6 address acl (NPB)

In a route map instance, matches IPv6 address conditions specified in an IPv6 ACL.

Syntax

match ipv6 address acl *acl-name*

no match ipv6 address acl *acl-name*

Parameters

acl-name

Specifies an IPv6 ACL name unique among all ACLs (Layer 2 and Layer 3). The name can from 1 through 63 characters in length and and must begin with an alphanumeric character. No special characters are allowed, except for the underscore and hyphen.

Modes

Route-map configuration mode

Usage Guidelines

A route-map stanza can contain only one **match { ip | ipv6 | mac |uda } address acl** statement.

The absence of a **match** statement is treated as "match any"; all traffic is forwarded according to the **set** statement.

Use the **no** form of this command to remove the match.

Examples

The following example creates an IPv6 ACL that permits traffic from specific sources and denies traffic from another source. The example then includes that ACL in a route-map stanza.

```
device# configure terminal
device(config)# ipv6 access-list extended acl6_NPB_01
device(conf-ip6acl-ext)# seq 10 permit ipv6 any host 2000::1 count
device(conf-ip6acl-ext)# seq 20 permit ipv6 any host 2000::2 count
device(conf-ip6acl-ext)# seq 30 deny ipv6 any host 2000::3 count
device(conf-ip6acl-ext)# exit
device(config)# route-map example2 permit 1
device(config-route-map-example2/permit/1)# match ipv6 address acl acl6_NPB_01
```

History

Release version	Command history
17s.1.01	This command was introduced.

match mac address acl (NPB)

In a route map instance, matches MAC address conditions specified in a Layer 2 ACL.

Syntax

match mac address acl *acl-name*

no match ip address acl *acl-name*

Parameters

acl-name

Specifies a Layer 2 ACL name unique among all ACLs (Layer 2 and Layer 3). The name can be up to 63 characters in length, and must begin with an alphanumeric character. No special characters are allowed, except for the underscore and hyphen.

Modes

Route-map configuration mode

Usage Guidelines

A route-map stanza can contain only one **match { ip | ipv6 | mac | uda } address acl** statement.

The absence of a **match** statement is treated as "match any"; all traffic is forwarded according to the **set** statement.

Use the **no** form of this command to remove the match.

Examples

The following example creates a Layer 2 (MAC) ACL that permits traffic from specific sources and denies traffic from another source. The example then includes that ACL in a route-map stanza.

```
device# configure terminal
device(config) mac access-list extended acl_4
device(conf-macl-ext)# permit host 00ab.0000.0001 any count
device(conf-macl-ext)# permit host 00ab.0000.0002 any count
device(conf-macl-ext)# deny host 00ab.0000.0003 any count
device(conf-macl-ext)# exit
device(config)# route-map example3 permit 1
device(config-route-map-example3/permit/1)# match mac address acl acl_4
```

History

Release version	Command history
17s.1.01	This command was introduced.

match uda address acl

In a route-map stanza, matches conditions specified in a user-developed ACL (UDA).

Syntax

match uda address acl *acl-name*

no match uda address acl *acl-name*

Parameters

acl-name

Specifies a UDA ACL name unique among all ACLs (Layer 2, Layer 3, and UDA). The name can from 1 through 63 characters in length and must begin with an alphanumeric character. No special characters are allowed, except for the underscore and hyphen.

Modes

Route-map configuration mode

Usage Guidelines

This command is supported only in NPB system mode.

A route-map stanza can contain only one **match { ip | ipv6 | mac | uda } address acl** statement.

The absence of a **match** statement is treated as "match any"; all traffic is forwarded according to the **set** statement.

Use the **no** form of this command to remove the match.

Examples

The following example creates a UDA ACL that permits traffic from a specific source IP and then includes that ACL in a route-map stanza.

```
device# configure terminal
device(config)# ip access-list standard aclNPB_01
device(conf-ipacl-std)# permit host 192.1.1.1 count
device(conf-ipacl-std)# exit
device(config)# route-map example1 permit 1
device(config-route-map-example1/permit/1)# match ip address acl acl_2
```

History

Release version	Command history
18s.1.00	This command was introduced.

npb policy route-map

On a physical or port-channel interface, applies a route map as Network Packet Broker (NPB) policy for all ingress traffic.

Syntax

npb policy route-map *route-map-name*

no npb policy route-map *route-map-name*

Command Default

No route map is applied as NPB policy.

Parameters

route-map-name

Specifies the route map. Values range from 1 through 63 ASCII characters.

Modes

Interface sub-type configuration mode

Usage Guidelines

This command is supported only for NPB. If the system mode is currently default, set it to NPB, using the **system-mode** command.

The **no** form of this command removes a route map previously applied for NPB policy.

Examples

The following example applies a route map to a physical interface.

```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# npb policy route-map npb_map1
```

The following example applies a route map to a port-channel.

```
device# configure terminal
device(config)# interface port-channel 10
device(config-Port-channel-10)# npb policy route-map npb_map1
```

History

Release version	Command history
17s.1.00	This command was introduced.
17s.1.01	This command was modified, with support added for port-channels.

profile ipv6-lookup

Specifies the IPv6 address lookup-mode.

Syntax

profile ipv6-lookup { **default** | **network-id** }

Command Default

The default IPv6 address lookup mode is by host ID.

Parameters

default

Resets the IPv6 address lookup-mode to host ID (bits 64–127 of the IPv6 address).

network-id

Changes the IPv6 address lookup-mode to network ID (bits 0–63 of the IPv6 address).

Modes

Hardware configuration mode

Usage Guidelines

ATTENTION
This is a disruptive command. In order for the last update of the profile configuration to take effect on a device, you must run the **copy running-config startup-config** command followed by the **reload system** command.

This command is available only in network packet broker (NBP) system-mode.

Each forwarded frame has a token with a 100-byte header buffer for storing header data. Fields copied into this token header-buffer are available for lookup (to make forwarding decisions). Because of this small buffer size, copying the entire 128-bit SIP and DIP addresses is not supported. You can configure the IPv6 lookup profile, specifying which half of SIP and DIP addresses are copied into the buffer:

- (Default) Host ID (bits 64-127)
- Network ID (bits 0-63)

The software does not prevent you from specifying a full 128-bit IPV6 address while configuring an IPv6 ACL. However, only the half configured in the IPv6 lookup profile is matched.

Before you change lookup-profile, you need to evaluate the impact of the change on current IPv6 ACLs.

There is not a **no** form of this command.

Examples

The following example changes the IPv6 lookup-profile from the default host-ID (bits 64-127) mode to network-id (bits 0-63) mode.

```
device# configure terminal
device(config)# hardware
device(config-hardware)# profile ipv6-lookup network-id
%Warning: To activate the new profile config, please run 'copy running-config startup-config' followed
by 'reload system'.
```

The following example restores the default IPv6 host-ID lookup-profile.

```
device# configure terminal
device(config)# hardware
device(config-hardware)# profile ipv6-lookup default
%Warning: To activate the new profile config, please run 'copy running-config startup-config' followed
by 'reload system'.
```

History

Release version	Command history
18s.1.00	This command was introduced.

route-map (NPB)

Creates a route map; for an existing route map, adds a stanza.

Syntax

route-map *name* { **permit** | **deny** } *stanza*

no route-map *name* { **permit** | **deny** } *stanza*

Parameters

name

Specifies the name of the route map. Names range from 1 through 63 ASCII characters in length.

permit

Enables the **set** statement within the specified stanza, as specified in the Usage Guidelines.

deny

Disables the **set** statement within the specified stanza, as specified in the Usage Guidelines.

stanza

Specifies the stanza ID. Valid values range from 1 through 65535.

Modes

Global configuration mode

Usage Guidelines

The following table describes the interactions between route-map **permit** and **deny** stanzas and **permit** and **deny** rules in ACLs applied to those stanzas by **match { mac | ip | ipv4 | uda } address acl** statements.

TABLE 31 Stanza and ACL **permit** and **deny** interactions

Stanza	ACL rule	Resulting TCAM action
Permit	Permit	The **set** statement or statements are applied.
Permit	Deny	Packets that match a **deny** keyword are denied from using the stanza **set** statement: • NPB: The packet is dropped. • PBR: The packet is routed as normal.
Deny	Permit	No action is taken: • NPB: The packet is dropped. • PBR: The packet is routed as normal.
Deny	Deny	If there are no subsequent matches, the packet is forwarded.

The **no** form of this command deletes a route-map stanza.

Examples

The following example enables the NPB route map for an Ethernet interface so that all ingress traffic from interface Ethernet 0/1 exits from port-channel 100. The absence of a **match** statement is treated as "match any"; all traffic is forwarded according to the **set** statement.

```
device# configure terminal
device(config)# route-map npb_map permit 10
device(config-route-map-npb_map/permit/10)# set interface port-channel 100
device(config-route-map-npb_map/permit/10)# exit
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# npb policy route-map npb_map
```

The following example configures ingress traffic from Ethernet 0/1 and port-channel 100 to egress Ethernet 0/5.

```
device# configure terminal
device(config)# route-map npb_map permit 10
device(config-route-map-npb_map/permit/10)# match ip address acl acl_2
device(config-route-map-npb_map/permit/10)# set interface ethernet 0/5
device(config-route-map-npb_map/permit/10)# exit
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# npb policy route-map npb_map
device(conf-if-eth-0/1)# exit
device(config)# interface port-channel 100
device(config-Port-channel-100)# npb policy route-map npb_map
```

The following example replicates traffic entering an interface to multiple egress interfaces.

```
device# configure terminal
device(config)# tvf-domain 10
device(config-tvf-domain-10)# description Sample TVF description
device(config-tvf-domain-10)# exit
device(config)# interface ethernet 0/5
device(conf-if-eth-0/5)# tvf-domain add 10
device(conf-if-eth-0/5)# exit
device(config)# interface port-channel 10
device(config-Port-channel-10)# tvf-domain add 10
device(config-Port-channel-10)# exit
device(config)# route-map npb_map1 permit 1
device(config-route-map-npb_map1/permit/1)# match ip address acl acl_2
device(config-route-map-npb_map1/permit/1)# set next-hop-tvf-domain 5
device(config-route-map-npb_map1/permit/1)# exit
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# npb policy route-map npb_map1
```

The following two-stanza route-map forwards frames according to the following conditions: Stanza 10 examines whether a frame matches the specified ACL; a match forwards the frame to port 0/21. If 0/21 is not up, the frame is forwarded to port-channel 666. If the Stanza 10 conditions are not met, Stanza 20 is examined. Because there is no match statement, it is considered a “match any”. All such traffic is forwarded to TVF domain 100—if it contains at least one interface and at least one of its member ports is up. If TVF 100 is not up, traffic is forwarded to interface 0/21, if up.

```
device# configure terminal
device(config)# route-map pbrTest permit 10
device(config-route-map-pbrTest/permit/10)# match ip address acl acl_pbr_05
device(config-route-map-pbrTest/permit/10)# set interface eth 0/21
device(config-route-map-pbrTest/permit/10)# set interface port-channel 666
device(config-route-map-pbrTest/permit/10)# exit

device(config)# route-map pbrTest permit 20
device(config-route-map-pbrTest/permit/20)# set next-hop-tvf-domain 100
device(config-route-map-pbrTest/permit/20)# set interface eth 0/21
```

History

Release version	Command history
17s.1.01	This command was introduced.

seq (deny/permit rules in UDAs)

Inserts filtering rules into user-defined ACLs (UDAs).

Syntax

seq *seq-value* { **deny** | **permit** } *uda-value-0 mask-0 uda-value-1 mask-1 uda-value-2 mask-2 uda-value-3 mask-3* [**count**] [**log**]

no seq *seq-value*

{ **deny** | **permit** } *uda-value-0 mask-0 uda-value-1 mask-1 uda-value-2 mask-2 uda-value-3 mask-3* [**count**] [**log**]

no { **deny** | **permit** } *uda-value-0 mask-0 uda-value-1 mask-1 uda-value-2 mask-2 uda-value-3 mask-3* [**count**] [**log**]

Command Default

The concluding, default rule is deny.

Parameters

seq
(Optional) Enables you to assign a sequence number to the rule. If you do not specify **seq** *seq-value*, the rule is added at the end of the list.

seq-value
Valid values range from 1 through 65535.

deny
Specifies rules to deny traffic.

permit
Specifies rules to permit traffic.

uda-value-0, *uda-value-1*, *uda-value-2*, *uda-value-3*
Specifies the hex values with which traffic is matched, at the offsets defined in the UDA profile.

mask-0, *mask-1*, *mask-2*, *mask-3*
Specifies hex-value masks for the UDA values.

count
Enables statistics for the rule.

log
Enables inbound logging for the rule.

Modes

UDA configuration mode

Usage Guidelines

Ths command is supported only in NPB system mode.

If there is no matching rule, the frame is permitted.

To delete a rule from an ACL, do the relevant of the following:

- If you know the rule number, enter **no seq** *seq-value*.
- If you do not know the rule number, type **no** followed by the full syntax without **seq** *seq-value*.

Examples

The following example creates a UDA and defines a permit rule.

```
device# configure terminal
device(config)# uda access-list extended uda_acl_1
device(conf-uda-acl-ext)# permit 0x112233 0xFFFFFF 0x0a0a0001 0xFFFFFFFF 0x0a0b0001 0xFFFFFFFF
0x11223344 0xFFFFFFFF
```

History

Release version	Command history
18s.1.00	This command was introduced.

set interface (NPB)

Specifies an egress interface for a route-map used to implement Network Packet Broker (NPB) policy.

Syntax

[**precedence** *precedence-value*] **set interface** { **ethernet** *slot* **/** *port* | **null0** | **port-channel** *number* } [**strip-vlan outer**] [**add-vlan outer** *vlan-id*]

no precedence *precedence-value*

no set interface { **ethernet** *slot* **/** *port* | **null0** | **port-channel** *number* } [**strip-vlan outer**] [**add-vlan outer** *vlan-id*]

Parameters

precedence
(Optional) Enables you to assign a precedence number to the set statement. If you do not specify **precedence** *precedence-value*, the statement is added at the end of the route map and a precedence number is automatically assigned to it.

precedence-value
Values range from 1 through 65535.

ethernet *slot* **/** *port*
Specifies an Ethernet interface. The slot number must be 0 if the switch does not contain slots.

null0
(Not implemented under NPB) Specifies the Null0 interface, dropping the packet.

port-channel *number*
Specifies a port-channel interface.

strip-vlan outer
Removes outer VLAN headers from the egressing packet.

add-vlan outer *vlan-id*
Adds an outer C-VLAN tag, specifying a VLAN ID. VLAN IDs range from 1 through 4090.

Modes

Route-map configuration mode

Usage Guidelines

This command is supported only under NPB system mode. If the system mode is default, set it to NPB, using the **system-mode** command.

The order of the set statements in a route-map is critical: In general, a match followed by a valid **set interface** or **set next-hop-tvf-domain** statement stops further processing. Specifying precedence values determines the order of statement processing. If you do not specify precedence values, they are automatically assigned as follows: The first set statement is assigned "precedence 10", the second is assigned "precedence 20", and so forth.

To display policy-map set-statement precedence values, run the **show running-config route-map** command. The results will make it easier for you to add additional set statements in the required order.

Adding a VLAN tag enables you to mark network traffic for custom processing downstream, such as application-specific filtering on an interconnected packet broker or special handling by the analytics application.

To delete a **set interface** statement from a route map, perform one of the following actions:

- If you know the precedence number, enter **no precedence** *precedence-value*.
- If you do not know the precedence number, type **no** followed by the full syntax without **precedence** *precedence-value*.

Examples

The following example configures ingress traffic from Ethernet 0/1 and port-channel 100 to egress Ethernet 0/5.

```
device# configure terminal
device(config)# route-map npb_map permit 10
device(config-route-map-npb_map/permit/10)# set interface ethernet 0/5
device(config-route-map-npb_map/permit/10)# exit
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# npb policy route-map npb_map
device(conf-if-eth-0/1)# exit
device(config)# interface port-channel 100
device(config-Port-channel-100)# npb policy route-map npb_map
```

History

Release version	Command history
17s.1.01	This command was introduced.
18s.1.00	This command was modifed to support the **add-vlan outer** *vlan-id* option.

set next-hop-tvf-domain

Specifies a Transparent VLAN Flooding (TVF) domain as the next hop for a Network Packet Broker (NPB) route map to support replication of traffic to multiple interfaces.

Syntax

[**precedence** *precedence-value*] **set next-hop-tvf-domain** *tvf-domain-ID* [**strip-vlan outer**] [**add-vlan outer** *vlan-id*]

no precedence *precedence-value*

no set next-hop-tvf-domain *tvf-domain-ID* [**strip-vlan outer**] [**add-vlan outer** *vlan-id*]

Command Default

TVF domain as the route map next hop is not configured.

Parameters

precedence

(Optional) Enables you to assign a precedence number to the set statement. If you do not specify **precedence** *precedence-value*, the statement is added at the end of the route map and a precedence number is automatically assigned to it.

precedence-value

Values range from 1 through 65535.

tvf-domain-ID

Specifies the ID of the TVF domain. Values are from 1 through 4096.

strip-vlan outer

Removes outer VLAN headers from the egressing packet.

add-vlan outer *vlan-id*

Adds an outer C-VLAN tag, specifying a VLAN ID. VLAN IDs range from 1 through 4090.

Modes

Route map configuration mode

Usage Guidelines

This command is supported only under NPB system mode. If the system mode is default, set it to NPB, using the **system-mode** command.

For load-balanced output when flooding, make sure that the TVF domain includes a port-channel.

The order of the set statements in a route-map is critical: In general, a match followed by a valid **set interface** or **set next-hop-tvf-domain** statement stops further processing. Specifying precedence values determines the order of statement processing. If you do not specify precedence values, they are automatically assigned as follows: The first set statement is assigned "precedence 10", the second is assigned "precedence 20", and so forth.

To display policy-map set-statement precedence values, run the **show running-config route-map** command. The results will make it easier for you to add additional set statements in the needed order.

To delete a **set next-hop-tvf-domain** statement from a route map, perform one of the following actions:

- If you know the precedence number, enter **no precedence** *precedence-value*.
- If you do not know the precedence number, type **no** followed by the full syntax without **precedence** *precedence-value*.

Examples

The following example configures—in a route map—a specified TVF domain as the next hop.

```
device# configure terminal
device(config)# route-map TVFtest permit 99
device(config-route-map-TVFtest/permit/99)# set next-hop-tvf-domain 5
```

History

Release version	Command history
17s.1.01	This command was introduced.
18s.1.00	This command was modifed to support the **add-vlan outer** *vlan-id* option.

show access-list

Displays ACL status information for a given network, protocol, and inbound/outbound direction.

Syntax

The following statement displays a summary of ACL statuses on the device:
show access-list { **ip** | **ipv6** | **mac** }

The following statement displays the status of an ACL on all device interfaces—applied to incoming or outgoing traffic:
show access-list { **ip** | **ipv6** | **mac** } *name* { **in** | **out** }

The following statements display the statuses on an interface of all ACLs applied to incoming or outgoing traffic:
show access-list interface { **ethernet** *slot* **/** *port* | **management** *port* | **port-channel** *index* } **in**

show access-list { **ve** *vlan_id* | **vlan** *vlan_id* } { **in** | **out** }

The following statements display the rules in an ACL applied to incoming or outgoing traffic on an interface:
show access-list mac *name* **interface** { **ethernet** *slot* **/** *port* | **port-channel** *index* } **in**

show access-list mac *name* **interface vlan** *vlan_id* { **in** | **out** }

show access-list { **ip** | **ipv6** } *name* **interface** { **ethernet** *slot* **/** *port* | **management** *port* | **port-channel** *index* } **in**

show access-list { **ip** | **ipv6** } *name* **interface ve** *vlan_id* { **in** | **out** }

The following statement displays the status of all receive ACLs (rACLs) applied to the device:
show access-list global in

The following statement displays details of a specified rACL applied to the device:
show access-list { **ip** | **ipv6** } *name* **global in**

Parameters

ip
Specifies the IPv4 Layer 3 network protocol.

ipv6
Specifies the IPv6 Layer 3 network protocol.

mac
Specifies the medium access control (MAC) Layer 2 network protocol.

in
Specifies incoming binding direction.

out
Specifies outgoing binding direction.

name
Specifies the ACL name.

interface
Filters by interface.

ethernet
Specifies a physical Ethernet interface.

slot
Specifies a valid slot number. Must be `0` if the switch does not contain slots.

port
Specifies a valid port number.

port-channel *index*
Specifies a port-channel interface.

ve *vlan_id*
Specifies a virtual Ethernet (VE) interface.

vlan *vlan_id*
Specifies a VLAN interface.

management *port*
Specifies a management interface.

global
Specifies Level 3 receive ACLs (rACLs), which are applied at device-level, rather than at interface-level.

Modes

Privileged EXEC mode

Usage Guidelines

You can show information for a specified ACL or only for that ACL on a specified interface. You can also display information for all ACLs bound to a specified physical interface, port-channel, VLAN or VE.

The command also displays information for receive-path ACLs.

Command Output

The **show access-list** command displays the following information:

Output field	Description
Active	The rule is active and implements the configured action.
Partial	The rule is partially programmed, with the configured action implemented in some cases. This is typically seen for logical interfaces like VLAN, which span multiple hardware resources.
In progress	The rule is currently being programmed into the hardware.
Inactive	The rule is inactive and is not programmed in the hardware. This is typically seen when the hardware resources limit is reached.

Examples

The following example displays the names of IPv4 ACLs applied to the device, interfaces to which they are applied, and the incoming/outgoing direction.

```
device# show access-list ip
Interface Ve 171
   Inbound access-list is not set
   Outbound access-list is IPV4_ACL_000 (From User)
Interface Ethernet 0/2
   Inbound switched access-list is IP_ACL_STD_EXAMPLE (From User)
   Outbound access-list is IP_ACL_EXT_EXAMPLE (From User)
```

The following example displays all interfaces on which an IPv4 ACL is applied in the outgoing direction.

```
device# show access-list ip IPV4_ACL_000 out
ip access-list IPV4_ACL_000 on Ve 171 at Egress (From User)
    seq 10 deny ip host 0.0.0.0 host 10.0.0.0 (Active)
```

The following example displays all interfaces on which an IPv6 ACL is applied in the incoming direction.

```
device# show access-list ipv6 distList in
ipv6 access-list distList on Ethernet 0/4 at Ingress (From User)
    seq 10 deny 2001:125:132:35::/64 (Active)
    seq 20 deny 2001:54:131::/64 (Active)
    seq 30 deny 2001:5409:2004::/64 (Active)
    seq 40 permit any (Active)
```

The following example displays all ACLs applied on a specified interface in the incoming direction.

```
device# show access-list interface ethernet 0/4 in
ipv6 access-list ipv6-std-acl on Ethernet 0/4 at Ingress (From User)
    seq 10 permit host 0:1::1 (Active)
    seq 20 deny 0:2::/64 (Active)
    seq 30 hard-drop any count (Active)
```

The following example displays IPv6 receive-path ACL information.

```
device# show access-list receive ipv6
ipv4 access-list extended ipv6-receive-acl-example
    seq 76 deny ip 10.10.95.10 0.0.0.0 any count (Active)

ipv6 access-list extended ipv6-receive-acl-example
    seq 10 deny ipv6 3001:2010:145:35::/64 any count (Active)
```

History

Release version	Command history
17s.1.00	This command was introduced.

show hardware profile

Displays details of the current active hardware profile. You can also display details for a specified TCAM, route-table, or IPv6-lookup profile.

Syntax

show hardware profile [**current**]

show hardware profile route-table { **default** | **ipv4-max-arp** | **ipv6-max-nd** | **multicast** | **multicast-snoop** | **user-defined** }

show hardware profile ipv6-lookup { **default** | **network-id** }

show hardware profile overlay-visibility { **default** | **endpoint** | **endpoint-vni** | **tunnel-vni** | **vni** }

(SLX 9140) **show hardware profile tcam** { **default** | **l2-acl-l3-iacl** | **l2-iacl-l3-acl** | **l2-l3-iacl-l2-iqos** | **l2-l3-iqos-l2-eacl** | **l2-l3-iqos-l2-iacl** | **l2-l3-iqos-l3-eacl** | **l2-l3-iqos-l3-iacl** | **user-defined** }

(SLX 9240) **show hardware profile tcam** { **default** | **l2-l3-iacl** | **l2-l3-iqos** | **l3-acl** | **l3-iacl-l2-eacl** | **l3-iacl-l2-iqos** | **l3-iqos-l2-iacl** | **user-defined** }

Parameters

current
Displays details of the current active profile.

ipv6-lookup
(NPB system-mode) Displays details of the IPv6 address lookup-mode.

default
Displays details of the default IPv6 address lookup-mode.

network-id
Displays details of the network-id IPv6 address lookup-mode.

overlay-visibility
(Default system-mode) Displays overlay-visibility profile information.

default
Matches on outer source IP and destination IP.

endpoint
Matches on outer source IP or destination IP.

endpoint-vni
Matches on outer source IP and VNI or destination IP and VNI.

tunnel-vni
Matches on outer source IP, destination IP, and VNI.

vni
Matches on VNI only.

route-table
(Default system-mode) Specifies hardware resources for route profiles.

default
Specifies IPv4/IPv6 resources for dual-stack operations.

ipv4-max-arp
Specifies resources for the maximum number of IPv4 ARP entries.

ipv6-max-nd
Specifies resources for the maximum number of IPv6 Neighbor Discovery entries.

multicast
Specifies resources for IP unicast dual-stack and IPv4 mulitcast.

multicast-snoop
Specifies resources for IP unicast dual-stack and multicast snooping.

user-defined
Specifies resources for a user-defined profile.

tcam (SLX 9140)
Specifies hardware resources for TCAM profiles.

default
Specifies resources with basic support for all applications.

l2-acl-l3-iacl
Specifies resources for ingress and egress Layer 2 ACLs; and ingress IPv4 and IPv6 ACLs.

l2-iacl-l3-acl
Specifies resources for ingress Layer 2 ACLs; and ingress and egress IPv4 and IPv6 ACLs.

l2-l3-iacl-l2-iqos
Specifies resources for ingress Layer 2, IPv4, and IPv6 ACLs; and ingress Layer 2 QoS.

l2-l3-iqos-l2-eacl
Specifies resources for ingress Layer 2, IPv4, and IPv6 QoS; and egress Layer 2 ACLs.

l2-l3-iqos-l2-iacl
Specifies resources for ingress Layer 2, IPv4, and IPv6 QoS; and ingress Layer 2 ACLs.

l2-l3-iqos-l3-eacl
Specifies resources for ingress Layer 2, IPv4, and IPv6 QoS; and egress IPv4 and IPv6 ACLs.

l2-l3-iqos-l3-iacl
Specifies resources for ingress Layer 2, IPv4, and IPv6 QoS; and ingress IPv4 and IPv6 ACLs.

user-defined
Specifies resources for a user-defined TCAM profile.

tcam (SLX 9240)
Specifies hardware resources for TCAM profiles.

default
Specifies resources with basic support for all applications.

l2-l3-iacl
Specifies resources for ingress Layer 2, IPv4, and IPv6 ACLs.

l2-l3-iqos
Specifies resources for ingress Layer 2, IPv4, and IPv6 QoS.

l3-acl
Specifies resources for ingress and egress IPv4 and IPv6 ACLs.

l3-iacl-l2-eacl
Specifies resources for egress Layer 2 ACLs; and ingress IPv4 and IPv6 ACLs.

l3-iacl-l2-iqos
Specifies resources for ingress Layer 2 QoS; and ingress IPv4 and IPv6 ACLs.

l3-iqos-l2-iacl
Specifies resources for ingress Layer 2 ACLs; and ingress IPv4 and IPv6 QoS.

user-defined
(Default system-mode only) Specifies resources for a user-defined TCAM profile.

Modes

Privileged EXEC mode

Usage Guidelines

Local hardware profile information can be obtained by means of the **current** keyword.

Route-table profiles are supported only in default system-mode, not in network packet broker (NPB) mode.

In NPB mode, the only TCAM profile supported is the default TCAM profile.

IPv6 address profiles are supported only NPB mode. For implementation details, refer to the **profile ipv6-lookup** topic.

Examples

(Default system-mode) The following example displays details of the current active profile.

```
device# show hardware profile current
 switch type: BR-SLX9240
 system mode: Default
                     current TCAM profile:    L2-L3-IACL
_____________________________________________________________
 MAC ACL Based QoS Policy Entries (Ingress):    0
         MAC Security ACL Entries (Ingress):    512
        MAC Policy Based forwarding entries:    0
IPV4 ACL Based QoS Policy Entries (Ingress):    0
IPV4 Policy Based Routing Entries (Ingress):    0
        IPV4 Security ACL Entries (Ingress):    0
IPV6 Policy Based Routing Entries (Ingress):    0
IPV6 ACL Based QoS Policy Entries (Ingress):    0
        IPV6 Security ACL Entries (Ingress):    0
          IP Security ACL Entries (Ingress):    512
  IP ACL Based QoS Policy Entries (Ingress):    0
          MAC Security ACL Entries (Egress):    0
  MAC ACL Based QoS Policy Entries (Egress):    0
         IPV4 Security ACL Entries (Egress):    0
 IPV4 ACL Based QoS Policy Entries (Egress):    0
         IPV6 Security ACL Entries (Egress):    0
 IPV6 ACL Based QoS Policy Entries (Egress):    0
           IP Security ACL Entries (Egress):    0
   IP ACL Based QoS Policy Entries (Egress):    0
_____________________________________________________________
                      current ROUTE profile:    DEFAULT
                              Maximum paths:    8
_____________________________________________________________
                        IPV4 neighbor cache:    10240
                        IPV6 neighbor cache:    4608
         IPV4 multicast multi-source groups:    512
         IPV4 multicast single-source group:    1024
                            Max IPV4 routes:    49152
                            Max IPV6 routes:    8192
                              Max next hops:    2048
                              L2 Forwarding:    29696
                      IPV4 multicast snoop:     1024
                      IPV6 multicast snoop:     512
                                    Vlans:      2048
                                  My MAC:       2048
```

The following example displays specific route-table information about resource allocation, facilitating management.

```
device# show hardware profile route-table ipv4-max-arp
 switch type: BR-SLX9240
 system mode: Default
                               ROUTE profile:    IPV4_MAX_ARP
_____________________________________________________________
                        IPV4 neighbor cache:    16384
                        IPV6 neighbor cache:    0
         IPV4 multicast multi-source groups:    0
         IPV4 multicast single-source group:    0
                            Max IPV4 routes:    49152
                            Max IPV6 routes:    8192
                              Max next hops:    2048
                              L2 Forwarding:    29696
                      IPV4 multicast snoop:     0
                      IPV6 multicast snoop:     0
                                    Vlans:      4096
                                  My MAC:       2048
_____________________________________________________________
```

The following example displays typical NPB information.

```
device# show hardware profile current

 switch type: BR-SLX9140
 system mode: NPB

                      current TCAM profile:    DEFAULT
_____________________________________________________________
        MAC Policy Based forwarding entries:    2048
    IP Policy Based Forwarding Entries (Ingress):    2048
    Flex Policy Based Forwarding Entries (Ingress):    1024
_____________________________________________________________
        current IPv6-Address Profile:    NETWORK-ID
_____________________________________________________________
    Valid IPv6 Source Address Bits: 0-63
    Valid IPv6 Destination Address Bits: 0-63
_____________________________________________________________
```

History

Release version	Command history
17s.1.00	This command was introduced.
17s.1.01	This command was modified to display default or network packet broker (NPB) system-mode.
18s.1.00	Under NPB system-mode, this command was modified to display default (host ID) or network-ID IPv6-address mode.

show inner-gtp-https

Displays a list of all interfaces on which the dropping of GPRS Tunneling Protocol (GTP) frames that encapsulate HTTPs packets is enabled.

Syntax

show inner-gtp-https

Modes

Privileged EXEC mode

Examples

The following example indicates that this feature is enabled on an Ethernet interface.

```
device# show inner-gtp-https
interface Ethernet 0/23
    deny inner-gtp-https
```

History

Release version	Command history
17s.1.02	This command was introduced.

show packet-encap-processing

Displays information about the interfaces on which header processing is enabled.

Syntax

show packet-encap-processing

Modes

Privileged EXEC mode

Usage Guidelines

Ths command is relevant only in NPB system mode.

Command Output

The **show packet-encap-processing** command displays the following information:

Output field	Description
Port	Displays the Ethernet slot and port.
Link	Displays "Up" or "Down".
Encapsulation	Displays the header-stripping type.
Status	If the header-stripping option is enabled on an interface, but not yet programmed in hardware—by a linecard or system reload—displays "Inactive". If the header-stripping option is enabled on an interface and is also programmed in hardware, displays "Active".

Examples

The following example displays a typical run of **show packet-encap-processing**.

```
show packet-encap-processing

Total number of packet-encap-processing interfaces: 6

-------------------------------------------------------
Port       Link     Encapsulation        Status
-------------------------------------------------------
eth 0/1    Down     802.1BR Stripping    Active
eth 0/2    Down     802.1BR Stripping    Active
eth 0/3    Down     VN-Tag Stripping     Active
eth 0/4    Up       VN-Tag Stripping     Active
eth 0/5    Up       VXLAN Stripping      Inactive
eth 0/6    Up       VXLAN Stripping      Inactive
```

History

Release version	Command history
18s.1.00	This command was introduced.

show route-map

Displays the route map configuration details.

Syntax

show route-map [*name*]

show route-map [**interface** { **ethernet** *slot* **/** *port* | **ve** *ve-number*]

Parameters

name

Specifies a route-map.

interface ethernet *slot* **/** *port*

Specifies a physical interface. If the device has no slots, the slot value must be 0.

ve *ve-number*

Specifies a Virtual Ethernet (VE) interface.

Modes

Privileged EXEC mode

Examples

The following command displays general route-map information.

```
device# show route-map
Interface Ethernet 0/6
 ip policy route-map route1
```

The following command displays the configured routing attributes of a specific route map.

```
device# show route-map route1
Interface Ethernet 0/6
ip policy route-map route1 permit 1 (Active)
    match ip address acl test1
    set ip next-hop 6.0.0.1 (selected)
    Policy routing matches: 1443 packets
```

The following command displays route-map configuration details for a specific interface.

```
device# show route-map interface ethernet 0/6
Interface Ethernet 0/6
ip policy route-map route1 permit 1 (Active)
    match ip address acl test1
    set ip next-hop 6.0.0.1 (selected)
    Policy routing matches: 1543 packets
```

History

Release version	Command history
17s.1.00	This command was introduced.

show running-config tvf-domain

Displays the running configuration of all defined Transparent VLAN Flooding (TVF) domains or of specified domains.

Syntax

show running-config tvf-domain [*tvf-domain-id*]

Parameters

tvf-domain-id

Specifies the ID of the TVF domain. Valid values are from 1 through 4096. To specify a range of domains, insert a hyphen (-) between the beginning and ending integers (for example, 5-16). To specify individual domains and ranges of domains, separate them with commas (for example: 1,5-7,55). Do not insert spaces after commas.

Modes

Privileged EXEC mode

Examples

The following command displays the names of all defined TVF domains.

```
device# show running-config tvf-domain
tvf-domain 1
description Sample TVF domain
!
tvf-domain 2
!
tvf-domain 3
!
tvf-domain 4
!
tvf-domain 5
!
```

History

Release version	Command history
17s.1.01	This command was introduced.
18s.1.00	This command was modified to display TVF-domain descriptions.

show statistics access-list

For a given network protocol and inbound/outbound direction, displays ACL statistical information. You can show statistics for a specified ACL or only for that ACL on a specified interface. You can also display statistical information for all ACLs bound to a specified device interface, VLAN or VE. You can also display statistical information for IPv4 or IPv6 receive-path ACLs.

Syntax

show statistics access-list interface { **ethernet** *slot* **/** *port* | **port-channel** *index* | **ve** *vlan_id* | **vlan** *vlan_id* } { **in** | **out** }

show statistics access-list { **ip** | **ipv6** } *name* **interface** [**ethernet** *slot* **/** *port* | **port-channel** *index* | **ve** *vlan_id*] { **in** | **out** }

show statistics access-list mac *name* **interface** [**ethernet** *slot* **/** *port* | **port-channel** *index* | **vlan** *vlan_id*] { **in** | **out** }

show statistics access-list global { **ip** | **ipv6** }

Parameters

interface
Filter by interface.

ethernet
Specifies a physical Ethernet interface.

slot
Specifies a valid slot number.

port
Specifies a valid port number.

port-channel *index*
Specifies a port-channel interface.

ve *vlan_id*
Specifies a virtual Ethernet (VE) interface.

vlan *vlan_id*
Specifies a VLAN interface.

in | **out**
Specifies the ACL binding direction (incoming or outgoing).

ip | **ipv6** | **mac**
Specifies the network protocol.

name
Specifies the ACL name.

global
Specifies IPv4 or IPv6 receive-path traffic.

Modes

Privileged EXEC mode

Usage Guidelines

Statistics are displayed only for rules that contain the **count** keyword.

Command Output

The **show statistics access-list** command displays the following information:

Output field	Description
Uncount	The counter resource is not allocated. This is typically seen if counting is not supported or if the hardware resources limit is reached.
Unwritten	The rule is inactive and is not programmed in the hardware. This is typically seen when the hardware resources limit is reached.

Examples

The following example displays inbound ACL statistics for a named IPv4 ACL.

```
device# show statistics access-list ip l3ext in
ip access-list l3ext Ethernet 0/8 in
seq 76 deny ip 10.10.75.10 0.0.0.0 any count log (795239 frames)
seq 77 hard-drop ip 10.10.75.10 0.0.0.0 10.10.11.0 0.0.0.255 count log (0 frames)
seq 78 hard-drop ip any 10.10.11.0 0.0.0.255 count log (0 frames)
seq 79 hard-drop ip any 10.10.0.0 0.0.255.255 count log (0 frames)
seq 80 hard-drop ip 10.10.75.10 0.0.0.0 any count log (0 frames)
seq 81 hard-drop ip 10.10.75.0 0.0.0.0 10.10.0.0 0.0.255.255 count log (0 frames)
seq 91 hard-drop ip any any count (0 frames)
seq 100 deny udp 10.10.75.0 0.0.0.255 10.10.76.0 0.0.0.255 count log (0 frames)
seq 1000 permit ip any any count log (0 frames)
```

The following example displays inbound ACL statistics for a specified interface. The ACL named ipv6-std-acl is applied on interface 0/1 to filter incoming routed traffic only.

```
device# show statistics access-list interface ethernet 0/1 in
ipv6 routed access-list ipv6-std-acl on Ethernet 0/1 at Ingress (From User)
    seq 10 permit host 0:1::1
    seq 20 deny 0:2::/64
    seq 30 deny any count (100 frames)
```

The following example displays inbound statistics for all ACLs bound to a specified VE interface.

```
device# show statistics access-list interface ve 3010 in
ipv6 access-list ip_acl_3 on Ve 3010 at Ingress (From User)
    seq 10 deny ipv6 2001:3010:131:35::/64 2001:1001:1234:1::/64 count (0 frames)
    seq 20 permit ipv6 2001:3010:131:35::/64 2001:3001:1234:1::/64
```

History

Release version	Command history
17s.1.00	This command was introduced.

strip-802-1br

Removes 802.1BR headers from incoming packets, for forwarding to the next processing port—for further filtering and forwarding.

Syntax

strip-802-1br

no strip-802-1br

Command Default

802.1BR header stripping is disabled.

Modes

Ethernet interface configuration mode

Usage Guidelines

This feature applies to both IPv4 and IPv6 traffic.

Enabling 802.1BR header-stripping on an interface requires the following settings:

- NPB system mode:

```
device# configure terminal
device(config)# hardware
device(config-hardware)# system-mode ?
Possible completions:
  [default]
  default   default mode
  npb       Network Packet Broker mode
device(config-hardware)# system-mode npb
```

- The default header-mode—802.1BR—must be enabled. If VN-Tag header-mode is enabled, restore 802.1BR mode:

```
device(config)# interface Ethernet 0/2
device(config-if-eth-0/2)# no allow-vn-tag
```

- Enable 802.1BR header stripping on the interface:

```
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# strip-802-1br
```

If a tunneled frame has an 802.1BR tag in the outer L2 header, VXLAN, NVGRE, ERSPAN, or L2-over-MPLS/pseudo-wire header-stripping also deletes the 802.1BR tag. (802.1BR tags in the inner L2 header are not supported.)

If both MPLS and 802.1BR header-stripping are configured, MPLS is always implemented. However, details vary between IP-over-MPLS and L2-over-MPLS regarding 802.1BR tags:

- Under IP-over-MPLS, only the MPLS labels are stripped; 802.1BR tags are not affected. The header diagram for this case is as follows:

IP-over-MPLS frame headers			
L2 (802.1BR tag)	MPLS	IPv4	TCP

- Under L2-over-MPLS (pseudo-wire), both the outer L2 (with 802.1BR tag) and the MPLS header are stripped. The header diagram for this case is as follows:

IP-over-MPLS frame (pseudo-wire) headers				
L2 (802.1BR tag)	MPLS	L2	IPv4	TCP

If interfaces with header-stripping enabled are included in a port-channel, the header stripping remains enabled, but only for traffic entering those interfaces.

The **no** form of this command disables 802.1BR header-stripping on the interface.

Examples

The following example enables 802.1BR header-stripping on the interface.

```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# strip-802-1br
```

The following example disables 802.1BR header-stripping on the interface.

```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# no strip-802-1br
```

History

Release version	Command history
18s.1.00	This command was introduced.

strip-erspan

Removes Encapsulated Remote Switch Port Analyzer (ERSPAN) headers from incoming packets, for forwarding to the next processing port—for further filtering and forwarding.

Syntax

strip-erspan

no strip-erspan

Command Default

ERSPAN header stripping is disabled.

Modes

Ethernet interface configuration mode

Usage Guidelines

This feature is relevant only under NPB system mode.

This feature applies to both IPv4 and IPv6 traffic.

For tunneled frames, header-stripping affects only the outer, encapsulation headers. For example:

- If both ERSPAN and VXLAN header-stripping are configured on the ingress interface, only the ERSPAN headers are stripped. (VXLAN header-stripping alone has no effect.) The header diagram for this case is as follows:

ERSPAN headers				VXLAN headers				Payload frame headers		
L2	IPv4	GRE	ERSPAN	L2	IPv4	GRE	ERSPAN	L2	IPv4	TCP

- If both ERSPAN and MPLS header-stripping are configured, only ERSPAN headers are stripped; MPLS labels are left untouched. The header diagram for this case is as follows:

ERSPAN headers				Payload frame headers			
L2	IPv4	GRE	ERSPAN	L2	MPLS	IPv4	TCP

If interfaces with header-stripping enabled are included in a port-channel, the header stripping remains enabled, but only for traffic entering those interfaces.

The **no** form of this command disables ERSPAN header-stripping on the interface.

Examples

The following example enables ERSPAN header-stripping on the interface.

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

The following example disables ERSPAN header-stripping on the interface.

```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# no strip-erspan
```

History

Release version	Command history
18s.1.00	This command was introduced.

strip-mpls

Removes Multi-Protocol Label Switching (MPLS) headers from incoming packets, for forwarding to the next processing port—for further filtering and forwarding.

Syntax

strip-mpls

no strip-mpls

Command Default

MPLS header stripping is disabled.

Modes

Ethernet interface configuration mode

Usage Guidelines

This feature is relevant only under NPB system mode.

This feature applies to both IPv4 and IPv6 traffic.

For tunneled frames, header-stripping affects only the outer, encapsulation headers. For example, if both VXLAN and MPLS header-stripping are configured, only VXLAN headers are stripped. The header diagram for this case is as follows:

VXLAN headers				Payload frame headers			
L2	IPv4	UDP	VXLAN	L2	MPLS	IPv4	TCP

If both MPLS and 802.1BR or VN-Tag header-stripping are configured, MPLS is always implemented. However, details vary between IP-over-MPLS and L2-over-MPLS regarding 802.1BR or VN-Tags:

- Under IP-over-MPLS, only the MPLS labels are stripped; 802.1BR or VN-Tags are not affected. The header diagram for this case is as follows:

IP-over-MPLS frame headers			
L2 (802.1BR or VN-Tag)	MPLS	IPv4	TCP

- Under L2-over-MPLS (pseudo-wire), both the outer L2 (with 802.1BR or VN-Tag) and the MPLS header are stripped. The header diagram for this case is as follows:

IP-over-MPLS frame (pseudo-wire) headers				
L2 (802.1BR or VN-Tag)	MPLS	L2	IPv4	TCP

If interfaces with header-stripping enabled are included in a port-channel, the header stripping remains enabled, but only for traffic entering those interfaces.

The **no** form of this command disables MPLS header-stripping on the interface.

Examples

The following example enables MPLS header-stripping on the interface.

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

The following example disables MPLS header-stripping on the interface.

```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# no strip-mpls
```

History

Release version	Command history
18s.1.00	This command was introduced.

strip-nvgre

Removes Network Virtualization using Generic Routing Encapsulation (NVGRE) headers from incoming packets, for forwarding to the next processing port—for further filtering and forwarding.

Syntax

strip-nvgre

no strip-nvgre

Command Default

NVGRE header-stripping is disabled.

Modes

Ethernet interface configuration mode

Usage Guidelines

This feature is relevant only under NPB system mode.

This feature applies to both IPv4 and IPv6 traffic.

For tunneled frames, header-stripping affects only the outer, encapsulation headers. For example:

- If both ERSPAN and NVGRE header-stripping are configured on the ingress interface, only the ERSPAN headers are stripped. (NVGRE header-stripping alone has no effect.) The header diagram for this case is as follows:

ERSPAN headers				NVGRE headers			Payload frame headers		
L2	IPv4	GRE	ERSPAN	L2	IPv4	NVGRE	L2	IPv4	TCP

- If both NVGRE and MPLS header-stripping are configured, only NVGRE headers are stripped: MPLS labels are left untouched. The header diagram for this case is as follows:

NVGRE headers			Payload frame headers			
L2	IPv4	NVGRE	L2	MPLS	IPv4	TCP

If interfaces with header-stripping enabled are included in a port-channel, the header stripping remains enabled, but only for traffic entering those interfaces.

The **no** form of this command disables NVGRE header-stripping on the interface.

Examples

The following example enables NVGRE header-stripping on the interface.

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

History

Release version	Command history
18s.1.00	This command was introduced.

strip-vn-tag

Removes Virtual NIC (VN)-tag headers from incoming packets, for forwarding to the next processing port—for further filtering and forwarding.

Syntax

strip-vn-tag

no strip-vn-tag

Command Default

VN-tag header stripping is disabled.

Modes

Ethernet interface configuration mode

Usage Guidelines

This feature applies to both IPv4 and IPv6 traffic.

Enabling VN-Tag header-stripping on an interface requires the following settings:

- NPB system mode. For example:

```
device# configure terminal
device(config)# hardware
device(config-hardware)# system-mode ?
Possible completions:
  [default]
  default   default mode
  npb       Network Packet Broker mode
device(config-hardware)# system-mode npb
```

- VN-Tag header-mode must be enabled. If 802.1BR mode:

```
device(config)# interface Ethernet 0/2
device(config-if-eth-0/2)# no allow-vn-tag
```

If a tunneled frame has a VN-Tag in the outer L2 header, VXLAN, NVGRE, ERSPAN, or L2-over-MPLS/pseudo-wire header-stripping also deletes the VN-Tag. (VN-Tags in the inner L2 header are not supported.)

If both MPLS and VN-Tag header-stripping are configured, MPLS is always implemented. However, details vary between IP-over-MPLS and L2-over-MPLS regarding VN-Tags:

- Under IP-over-MPLS, only the MPLS labels are stripped; VN-Tags are not affected. The header diagram for this case is as follows:

IP-over-MPLS frame headers			
L2 (VN-Tag)	MPLS	IPv4	TCP

- Under L2-over-MPLS (pseudo-wire), both the outer L2 (with VN-Tag) and the MPLS header are stripped. The header diagram for this case is as follows:

IP-over-MPLS frame (pseudo-wire) headers				
L2 (VN-Tag)	MPLS	L2	IPv4	TCP

If interfaces with header-stripping enabled are included in a port-channel, the header stripping remains enabled, but only for traffic entering those interfaces.

The **no** form of this command disables VN-tag header-stripping on the interface.

Examples

The following example enables VN-tag header-stripping on the interface.

```
device# configure terminal
device(config)# interface ethernet 0/2
device(conf-if-eth-0/2)# strip-vn-tag
```

History

Release version	Command history
18s.1.00	This command was introduced.

strip-vxlan

Removes Virtual Extensible LAN (VXLAN) headers from incoming packets, for forwarding to the next processing port—for further filtering and forwarding.

Syntax

strip-vxlan

no strip-vxlan

Command Default

VXLAN header-stripping is disabled.

Modes

Ethernet interface configuration mode

Usage Guidelines

This feature is relevant only under NPB system mode.

On an interface, you can enable both VXLAN and NVGRE header-stripping. (VXLAN and NVGRE are mutually exclusive flows.)

This feature applies to both IPv4 and IPv6 traffic.

The final Frame Check Sequence (FCS) is updated with a recalculated CRC.

If interfaces with header-stripping enabled are included in a port-channel, the header stripping remains enabled, but only for traffic entering those interfaces.

For tunneled frames, header-stripping affects only the outer, encapsulation headers. For example:

- If both ERSPAN and VXLAN header-stripping are configured on the ingress interface, only the ERSPAN headers are stripped. The header diagram for this case is as follows:

ERSPAN headers				VXLAN headers				Payload frame headers		
L2	IPv4	GRE	ERSPAN	L2	iIPv4	UPD	VXLAN	L2	IPv4	TCP

- If both VXLAN and MPLS header-stripping are configured, only VXLAN headers are stripped. The header diagram for this case is as follows:

VXLAN headers				Payload frame headers			
L2	IPv4	UDP	VXLAN	L2	MPLS	IPv4	TCP

The **no** form of this command disables VXLAN header-stripping on the interface.

Examples

The following example enables VXLAN header-stripping on the interface.

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

History

Release version	Command history
18s.1.00	This command was introduced.

system-mode

Sets the system mode.

Syntax

system-mode { **default** | **npb** }

Parameters

default

Specifies the default system mode.

npb

Specifies the Network Packet Broker (NPB) system mode.

Modes

Hardware configuration mode

Usage Guidelines

In NPB mode, Layer 2 and Layer 3 forwarding, protocols, and services such as SPAN and SFLOW are not supported. Extreme recommends not to use any of these configurations in NPB mode. If these features are required, use default mode.

Examples

The following example indicates that the current mode is default. The value displayed within brackets ([ ]) is the current mode.

NOTE

The **show running-config hardware** command also displays the current mode.

```
device# configure terminal
device(config)# hardware
device(config-hardware)# system-mode ?
Possible completions:
  [default]
  default   default mode
  npb       Network Packet Broker mode
```

The following example sets the NPB system mode and reloads the system.

```
device# configure terminal
device(config)# hardware
device(config-hardware)# system-mode npb
%Warning: To activate the new system-mode config, please reboot the system using 'reload system'.
device(config-hardware)# exit
device(config)# exit
device# reload system
Warning: This operation will cause the chassis to reboot and requires all existing telnet, secure
telnet and SSH sessions to be restarted.
Unsaved configuration will be lost. Please run `copy running-config startup-config` to save the current
configuration if not done already.
Are you sure you want to reboot the chassis [y/n]? y <Enter>
```

The following example resets the default system mode.

```
device# configure terminal
device(config)# hardware
device(config-hardware)# system-mode default
%Warning: To activate the new system-mode config, please reboot the system using 'reload system'.
```

History

Release version	Command history
17s.1.00	This command was introduced.

tvf-domain

Creates one or more Transparent VLAN Flooding (TVF) domains.

Syntax

tvf-domain *tvf-domain-id*

no tvf-domain *tvf-domain-id*

Parameters

tvf-domain-id

Specifies the ID of the TVF domain. Valid values are from 1 through 4096. To specify a range of domains, insert a hyphen (-) between the beginning and ending integers (for example, 5-16). To specify individual domains and ranges of domains, separate them with commas (for example: 1,5-7,55). Do not insert spaces after commas.

Modes

Global configuration mode

Usage Guidelines

This command is available only in NPB system-mode.

TVF forwards packets without CPU intervention—such as MAC learning or MAC destination lookups—enabling line-rate traffic forwarding.

The maximum number of supported TVF domains is 1024.

Under Network Packet Brokering (NPB), TVF domains are required for the traffic replication feature.

The **no** form of this command deletes a specified TVF domain.

Examples

The following example creates a TVF domain.

```
device# configure terminal
device(config)# tvf-domain 10
device(config-tvf-domain-10)# description Sample TVF domain
```

History

Release version	Command history
17s.1.01	This command was introduced.
18s.1.00	The example was modified to include a TVF-domain description.

tvf-domain (interface)

Assigns and removes Transparent VLAN Flooding (TVF) domains from a physical or port-channel interface.

Syntax

tvf-domain { **add** *tvf-domain-id* | **all** | **except** *tvf-domain-id* | **none** | **remove** *tvf-domain-id* }

Command Default

No TVF domains are assigned to the interface.

Parameters

add *tvf-domain-id*

Assigns one or more TVF domains to the interface. To specify a range of IDs, insert a hyphen between the beginning and ending integers (for example, 5-16). To specify individual IDs and ranges of IDs, separate them with commas (for example: 1,5-7,55). Do not insert spaces after commas. You can enter a maximum of 253 characters.

all

Assigns all defined TVF domains to the interface.

except *tvf-domain-id*

Assigns all TVF domains to the interface, except for those specified. To specify a range of IDs, insert a hyphen between the beginning and ending integers (for example, 5-16). To specify individual IDs and ranges of IDs, separate them with commas (for example: 1,5-7,55). Do not insert spaces after commas. You can enter a maximum of 253 characters.

none

Removes all TVF domains assigned to the interface.

remove *tvf-domain-id*

Removes one or more TVF domains from the interface. To specify a range of IDs, insert a hyphen between the beginning and ending integers (for example, 5-16). To specify individual IDs and ranges of IDs, separate them with commas (for example: 1,5-7,55). Do not insert spaces after commas. You can enter a maximum of 253 characters.

Modes

Interface sub-type configuration mode

Usage Guidelines

Under Network Packet Brokering (NPB), TVF domains are required for the traffic replication feature.

This command is available only in NPB mode.

You can create as many as 4096 TVF domains. Domain members can be tagged and untagged ports. There is no software limitation on the number of member ports.

Examples

The following example assigns a TVF domain that you create to a physical interface.

```
device# configure terminal
device(config)# tvf-domain 10
device(config-tvf-domain-10)# description Sample TVF domain
device(config-tvf-domain-10)# exit
device(config)# interface ethernet 0/30
device(conf-if-eth-0/30)# tvf-domain add 10
device(conf-if-eth-0/30)# no shut
```

History

Release version	Command history
17s.1.01	This command was introduced.

uda access-list

Creates a user-defined ACL (UDA). UDAs offer greater flexibility than other ACLs in defining deny and permit rules. This flexibility is required for certain Network Packet Broker (NPB) scenarios.

Syntax

uda access-list extended *acl-name*

no uda access-list extended *acl-name*

Command Default

No UDA is defined.

Parameters

extended

Specifies an extended ACL. Extended ACLs support source and destination addresses, as well as other parameters. UDAs cannot be standard ACLs, which filter by source address only.

acl-name

Specifies an ACL name unique among all ACLs (Layer 2, Layer 3, and UDAs). The name can be up to 63 characters in length, and must begin with an alphanumeric character. No special characters are allowed, except for the underscore and hyphen.

Modes

Global configuration mode

Usage Guidelines

Ths command is supported only in NPB system mode.

A UDA starts functioning on an interface—for aggregation, replication, or forwarding—only if the following flow is implemented:

- Create a UDA profile, using the **uda-key profile** command.
- For the profile, specify the header types in the expected packet structure, using the **flow** command.
- For the profile, specify the header fields to match, using the **uda-key** command.
- Apply the profile to the interface, using the **uda-profile-apply** command.
- Create a UDA, using the **uda access-list** command.
- Create one or more permit or deny rules in the UDA, using the [**seq** *seq-value*] { **deny** | **permit** } command.
- Create a route-map, using the **route-map** command.
- Apply the UDA to the route-map, using the **match uda address acl** command.
- In the route-map, specify the egress interface, using the **set interface** or **set next-hop-tvf-domain** command.
- On a physical or port-channel interface, apply the route map to the ingress interface, using the **npb policy route-map** command.

To delete a UDA that is not referenced by any route-map, use the **no** form of this command.

Examples

The following example creates a UDA.

```
device# configure terminal
device(config)# uda access-list extended uda_01
```

The following example deletes a UDA.

```
device# configure terminal
device(config)# no uda access-list extended uda_01
```

History

Release version	Command history
18s.1.00	This command was introduced.

uda-key

In a user-defined ACL (UDA)-profile, specifies up to four header fields for matching.

Syntax

uda-key0 header{0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 } *header-field*

uda-key1 header{0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 } *header-field*

uda-key2 header{0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 } *header-field*

uda-key3 header{0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 } *header-field*

no uda-key0 header{0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 } *header-field*

no uda-key1 header{0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 } *header-field*

no uda-key2 header{0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 } *header-field*

no uda-key3 header{0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 } *header-field*

Command Default

No UDA key is specified.

Parameters

uda-key{0 | 1 | 2 | 3 }
: Specifies 1 through 4 UDA keys. You assign a header type and field to each key.

header{0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 }
: Specifies a header type defined in the **flow** command.

header-field
: Specifies a supported field for one of the header types defined in the **flow** command. For fields supported for each header type, refer to Usage Guidelines.

Modes

UDA-profile configuration mode

Usage Guidelines

Ths command is supported only in NPB system mode.

You are not required to specify all four UDA keys. Keys not specified are programmed as match always (don't care).

You can specify fields from a header type in more than one UDA key.

A UDA starts functioning on an interface—for aggregation, replication, or forwarding—only if the following flow is implemented:

- Create a UDA profile, using the **uda-key profile** command.
- For the profile, specify the header types in the expected packet structure, using the **flow** command.

- For the profile, specify the header fields to match, using the **uda-key** command.
- Apply the profile to the interface, using the **uda-profile-apply** command.
- Create a UDA, using the **uda access-list** command.
- Create one or more permit or deny rules in the UDA, using the [**seq** *seq-value*] { **deny** | **permit** } command.
- Create a route-map, using the **route-map** command.
- Apply the UDA to the route-map, using the **match uda address acl** command.
- In the route-map, specify the egress interface, using the **set interface** or **set next-hop-tvf-domain** command.
- On a physical or port-channel interface, apply the route map to the ingress interface, using the **npb policy route-map** command.

To delete a UDA key, use the **no** form of this command.

To modify a key, delete the current key and define a new one.

The following tables display the fields supported for each header type:

TABLE 32 ETHERNET (Ethernet header)

Field	Width (Bits)	Description
HAS_INNER_TAG	1	Packet has inner tag?
HAS_OUTER_TAG	1	Packet has outer tag?
HAS_8021BR_TAG	1	Packet has 802.1BR tag?
HAS_VNTAG	1	Packet has VN tag?
ETHER_TYPE	16	EtherType
INNER_VID	16	PCP or DEI or VLAN ID
OUTER_VID	16	PCP or DEI or VLAN ID
8021BR_TAG	32	802.1BR tag
VNTAG	32	VN tag
MAC_SA_16_47	32	Bits 16–47 of SA MAC
MAC_SA_0_15	16	Bits 0–15 of SA MAC
MAC_DA_32_47	16	Bits 32–47 of DA MAC
MAC_DA_0_31	32	Bits 0–31 of DA MAC

TABLE 33 TUN_ETHERNET (Inner Ethernet header in tunneled frames)

Field	Width (Bits)	Description
HAS_INNER_TAG	1	Packet has inner tag?
HAS_OUTER_TAG	1	Packet has outer tag?
ETHER_TYPE	16	EtherType
INNER_VID	16	PCP or DEI or VLAN ID
OUTER_VID	16	PCP or DEI or VLAN ID
MAC_SA_16_47	32	Bits 16–47 of SA MAC
MAC_SA_0_15	16	Bits 0–15 of SA MAC
MAC_DA_32_47	16	Bits 32–47 of DA MAC
MAC_DA_0_31	32	Bits 0–31 of DA MAC

TABLE 34 IPV4 (IPv4 header)

Field	Width (Bits)	Description
DIP	32	Destination IP
SIP	32	Source IP
PROTOCOL	8	IP protocol
TOTAL_LENGTH	16	Total length
TOS	8	Type of service (DSCP & ECN)
ECN	2	ECN
DSCP	6	DSCP

TABLE 35 IPV6 (IPv6 header)

Field	Width (Bits)	Description
NEXT_HEADER	8	Next header
TOTAL_LENGTH	16	Total length
TRAFFIC_CLASS	8	Traffic class (DSCP & ECN)
ECN	2	ECN
DSCP	6	DSCP
DIP1	32	Bits 32-63 or 96-127 of Destination IP
DIP0	32	Bits 0-31 or 64-95 of Destination IP
SIP1	32	Bits 32-63 or 96-127 of Source IP
SIP0	32	Bits 0-31 or 64-95 of Source IP

TABLE 36 ARP (ARP header)

Field	Width (Bits)	Description
TARGET_IP	32	Target IP
TARGET_HW_ADDR_16_47	32	Bits 16-47 of target HW address
TARGET_HW_ADDR_0_15	16	Bits 0-15 of target HW address
SENDER_IP_16_31	16	Bits 16-31 of sender IP
SENDER_IP_0_15	16	Bits 0-15 of sender IP
SENDER_HW_ADDR_32_47	16	Bits 32-47 of sender HW address
SENDER_HW_ADDR_0_31	32	Bits 0-31 of sender HW address

TABLE 37 TCP (TCP header)

Field	Width (Bits)	Description
FLAGS	8	TCP flags
PORTS	32	Source and destination port
DST_PORT	16	Destination port
SRC_PORT	16	Source port

TABLE 38 UDP (UDP header)

Field	Width (Bits)	Description
PORTS	32	Source and destination port
DST_PORT	16	Destination port

TABLE 38 UDP (UDP header) (continued)

Field	Width (Bits)	Description
SRC_PORT	16	Source port

TABLE 39 SCTP (SCTP header)

Field	Width (Bits)	Description
PORTS	32	Source and destination port
DST_PORT	16	Destination port
SRC_PORT	16	Source port

TABLE 40 IGMP (IGMP header)

Field	Width (Bits)	Description
TYPE	8	Type

TABLE 41 ICMP (ICMP header)

Field	Width (Bits)	Description
TYPE_CODE	16	ICMP type and code

TABLE 42 ICMPV6 (ICMPv6 header)

Field	Width (Bits)	Description
TYPE_CODE	16	ICMPv6 type and code

TABLE 43 MPLS (MPLS header)

Field	Width (Bits)	Description
LABEL0	24	Label 0 (Label, EXP & S-Bit)
LABEL1	24	Label 1 (Label, EXP & S-Bit)
LABEL2	24	Label 2 (Label, EXP & S-Bit)
LABEL3	24	Label 3 (Label, EXP & S-Bit)

TABLE 44 GRE (GRE header)

Field	Width (Bits)	Description
IS_NVGRE	1	Packet has NVGRE encapsulation?
IS_ERSPAN	1	Packet has ERSPAN encapsulation?
IS_L2	1	Is payload L2?
IS_IPV4	1	Is payload IPv4?
IS_IPV6	1	Is payload IPv4?
FLAGS	8	First byte of the GRE header
KEY_24_31	8	Bits 24–31 of GRE key
KEY_0_23	24	Bits 0–23 of GRE key
TNI	24	NVGRE tenant network ID

TABLE 45 VXLAN (VXLAN header)

Field	Width (Bits)	Description
VNI	24	VXLAN network identifier

TABLE 46 GTP (GTP header)

Field	Width (Bits)	Description
TEID	32	GTP tunnel endpoint ID

TABLE 47 PAYLOAD4 (4 bytes)

Field	Width (Bits)	Description
WORD0	32	Word 0 in the payload

TABLE 48 PAYLOAD8 (8 bytes)

Field	Width (Bits)	Description
WORD0	32	Word 0 in the payload
WORD1	32	Word 1 in the payload

TABLE 49 PAYLOAD16 (16 bytes)

Field	Width (Bits)	Description
WORD0	32	Word 0 in the payload
WORD1	32	Word 1 in the payload
WORD2	32	Word 2 in the payload
WORD3	32	Word 3 in the payload

TABLE 50 PAYLOAD32 (32 bytes)

Field	Width (Bits)	Description
WORD0	32	Word 0 in the payload
WORD1	32	Word 1 in the payload
WORD2	32	Word 2 in the payload
WORD3	32	Word 3 in the payload
WORD4	32	Word 4 in the payload
WORD5	32	Word 5 in the payload
WORD6	32	Word 6 in the payload
WORD7	32	Word 7 in the payload

Examples

The following example configures a UDA profile, creates a flow, and specifies header types and fields.

```
device(config)# uda-key profile fkp1
device(conf-uda-profile-fkp1)# flow header0 ETHERNET header1 IPV4 header2 UDP header3 PAYLOAD32
device(conf-uda-profile-fkp1)# uda-key0 header0 ETHER_TYPE
device(conf-uda-profile-fkp1)# uda-key1 header1 PROTOCOL
device(conf-uda-profile-fkp1)# uda-key2 header2 DST_PORT
device(conf-uda-profile-fkp1)# uda-key3 header3 WORD1
```

History

Release version	Command history
18s.1.00	This command was introduced.

uda-key profile

Creates a user-defined ACL (UDA) profile, which specifies packet header fields for filtering.

Syntax

uda-key profile *profile-name*

no uda-key profile *profile-name*

Command Default

No UDA profile is defined.

Parameters

profile-name

Specifies a unique UDA-profile name. Valid names range from 1 through 63 characters.

Modes

Global configuration mode

Usage Guidelines

Ths command is supported only in NPB system mode.

A UDA starts functioning on an interface—for aggregation, replication, or forwarding—only if the following flow is implemented:

- Create a UDA profile, using the **uda-key profile** command.
- Configure the profile, using the **flow** and **uda-key** commands.
- Apply the profile to the interface, using the **uda-profile-apply** command.
- Create a UDA, using the **uda access-list** command.
- Create one or more permit or deny rules in the UDA, using the [**seq** *seq-value*] { **deny** | **permit** } command.
- Create a route-map, using the **route-map** command.
- Apply the UDA to the route-map, using the **match uda address acl** command.
- In the route-map, specify the egress interface, using the **set interface** or **set next-hop-tvf-domain** command.
- On a physical or port-channel interface, apply the route map to the ingress interface, using the **npb policy route-map** command.

To delete a UDA profile that is not applied to any interfaces, use the **no** form of this command.

Examples

The following example defines a UDA profile.

```
device# configure terminal
device(config)# uda-key profile prof_01
device(config-uda-key)#
```

The following example deletes a UDA profile.

```
device# configure terminal
device(config)# no uda-key profile prof_01
```

History

Release version	Command history
18s.1.00	This command was introduced.

uda-profile-apply

Applies a UDA profile to an Ethernet or port-channel interface.

Syntax

uda-profile-apply *profile-name*

no uda-profile-apply *profile-name*

Command Default

No UDA profile is applied.

Parameters

profile-name

Specifies a unique UDA-profile name. Valid names range from 1 through 63 characters.

Modes

Ethernet interface configuration mode

Port-channel configuration mode

Usage Guidelines

You can apply a UDA profile only to an Ethernet or to a port-channel interface.

It is not sufficient for a UDA to be applied to an interface—in a route-map. Unless a UDA-key profile is also applied to that interface, traffic on that interface is programmed as match-always.

To remove a UDA profile from an interface, use the **no** form of this command.

Examples

The following example applies a UDA profile to an Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# uda-profile-apply prof_01
```

The following example removes a UDA profile from an Ethernet interface.

```
device# configure terminal
device(config)# interface ethernet 0/1
device(conf-if-eth-0/1)# no uda-profile-apply prof_01
```

History

Release version	Command history
18s.1.00	This command was introduced.