Chapter 24

ACLs and Route Maps

The switch uses rule-based lists to control packet access to ports and to select routes for redistribution to routing domains defined by dynamic routing protocols. This chapter describes the construction of access control lists (ACLs), prefix lists, and route maps.

This chapter includes the following sections:

- Section 24.1: ACL, Service ACL, Route Map, Prefix List, and RACL Divergence Introduction
- Section 24.2: Access Control Lists
- Section 24.3: Service ACLs
- Section 24.4: RACL Sharing on SVIs
- Section 24.5: Route Maps
- Section 24.6: Prefix Lists
- Section 24.7: ACL, Route Map, and Prefix List Commands

24.1 ACL, Service ACL, Route Map, Prefix List, and RACL Divergence Introduction

Access control lists (ACLs), Service ACLs, route maps, and prefix lists are all processed in order, beginning with the first rule and proceeding until a match is encountered.

An access control list (ACL) is a list of rules that control the inbound flow of packets into Ethernet interfaces, subinterfaces, and port channel interfaces or the switch control plane. The switch supports the implementation of a wide variety of filtering criteria including IP and MAC addresses, TCP/UDP ports with include/exclude options without compromising its performance or feature set. Filtering syntax is industry standard.

A Service ACL is an ACL applied by a control-plane process to control connections to, or packets processed by, the agent process.

A route map is a list of rules that control the redistribution of IP routes into a protocol domain on the basis of such criteria as route metrics, access control lists, next hop addresses, and route tags. Route maps can also alter parameters of routes as they are redistributed.

A prefix list is a list of rules that defines route redistribution access for a specified IP address space. Route maps often use prefix lists to filter routes.

The RACL divergence optimizes the usage of hardware resources occupied on each forwarding ASIC by installing ACLs only on the hardware components corresponding to the member interfaces belonging to the SVIs on which ACL is applied. Hence, saving the hardware resources used and enables RACLs to scale-up to a larger configuration. The show commands are used to display the interface mapping, TCAM entries, and TCAM utilization information.
24.2 Access Control Lists

These sections describe access control lists:

- Section 24.2.1: ACL Description
- Section 24.2.2: ACL Configuration
- Section 24.2.3: Applying ACLs

24.2.1 ACL Description

This section describes ACL composition and function. The switch supports the following ACL types:

- IPv4
- IPv6
- Standard IPv4
- Standard IPv6
- MAC

24.2.1.1 ACL Structure

An ACL is an ordered list of rules that defines access restrictions for the entities (the control plane, or an interface) to which it is applied. ACLs are also used by route maps to select routes for redistribution into specified routing domains.

ACL rules specify the data to which packet contents are compared when filtering data.

- The interface forwards packets that match all commands in a permit rule.
- The interface drops packets that match all commands in a deny rule.
- The interface drops packets that do not match at least one rule.

Upon its arrival at an interface, a packet's fields are compared to the first rule of the ACL applied to the interface. Packets that match the rule are forwarded (permit rule) or dropped (deny rule). Packets that do not match the rule are compared to the next rule in the list. This process continues until the packet either matches a rule or the rule list is exhausted. The interface drops packets not matching a rule.

The sequence number designates the rule's placement in the ACL.

24.2.1.2 ACL Rules

ACL rules consist of a command list that is compared to inbound packet fields. When all of a rule's criteria match a packet's contents, the interface performs the action specified by the rule.

The set of available commands depend on the ACL type and the specified protocol within the rule. The following is a list of commands available for supported ACL types

### IPv4 ACL Rule Parameters

All rules in IPv4 ACLs include the following criteria:

- **Protocol**: The packet's IP protocol. Valid rule inputs include:
  - Protocol name for a limited set of common protocols.
  - Assigned protocol number for all IP protocols.
- **Source Address**: The packet's source IPv4 address. Valid rule inputs include:
  - a subnet address (CIDR or address-mask). Discontiguous masks are supported.
• a host IP address (dotted decimal notation).
• any to denote that the rule matches all source addresses.
• **Destination Address:** The packet’s destination IP address. Valid rule inputs include:
  • a subnet address (CIDR or address-mask). Discontiguous masks are supported.
  • a host IP address (dotted decimal notation).
  • any to denote that the rule matches all destination addresses.

All rules in IPv4 ACLs **may** include the following criteria:

• **Fragment:** Rules filter on the fragment bit.
• **Time-to-live:** Compares the TTL (time-to-live) value in the packet to a specified value. Valid in ACLs applied to the control plane. Validity in ACLs applied to the data plane varies by switch platform. Comparison options include:
  • **Equal:** Packets match if packet value equals statement value.
  • **Greater than:** Packets match if packet value is greater than statement value.
  • **Less than:** Packets match if packet value is less than statement value.
  • **Not equal:** Packets match if packet value does not equal statement value.

The availability of the following optional criteria depends on the specified protocol:

• **Source Ports / Destination Ports:** A rule filters on ports when the specified protocol supports IP address-port combinations. Rules provide one of these port filtering values:
  • any denotes that the rule matches all ports.
  • A list of ports that matches the packet port. Maximum list size is 10 ports.
  • Negative port list. The rule matches any port not in the list. Maximum list size is 10 ports.
  • Integer (lower bound): The rule matches any port with a number larger than the integer.
  • Integer (upper bound): The rule matches any port with a number smaller than the integer.
  • Range integers: The rule matches any port whose number is between the integers.
• **Flag bits:** Rules filter TCP packets on flag bits.
• **Message type:** Rules filter ICMP type or code.
• **Tracked:** Matches packets in existing ICMP, UDP, or TCP connections. Valid in ACLs applied to the control plane. Validity in ACLs applied to the data plane varies by switch platform.

**IPv6 ACL Rule Parameters**

**Note**

When calculating the size of ACLs, be aware that Arista switches install four rules in every IPv6 ACL so that ICMPv6 neighbor discovery packets bypass the default drop rule.

All rules in IPv6 ACLs include the following criteria:

• **Protocol:** All rules filter on the packet’s IP protocol field. Rule input options include:
  • Protocol name for a limited set of common protocols.
  • Assigned protocol number for all IP protocols.
• **Source Address:** The packet’s source IPv6 address. Valid rule inputs include:
  • an IPv6 prefix (CIDR). Discontiguous masks are supported.
  • a host IP address (dotted decimal notation).
  • any to denote that the rule matches all addresses.
• **Destination Address**: The packet’s destination IP address. Valid rule inputs include:
  • a subnet address (CIDR or address-mask). Discontiguous masks are supported.
  • a host IP address (dotted decimal notation).
  • any to denote that the rule matches all addresses.

All rules in IPv6 ACLs may include the following criteria:

• **Fragment**: Rules filter on the fragment bit.
• **Hop**: Compares the packet’s hop-limit value to a specified value. Comparison options include:
  • Equal: Packets match if packet value equals statement value.

The availability of the following optional criteria depends on the specified protocol:

• **Source Ports / Destination Ports**: A rule filters on ports when the specified protocol supports IP address-port combinations. Rules provide one of these port filtering values:
  • any denotes that the rule matches all ports.
  • A list of ports that matches the packet port. Maximum list size is 10 ports.
  • Negative port list. The rule matches any port not in the list. Maximum list size is 10 ports.
  • Integer (lower bound): The rule matches any port with a number larger than the integer.
  • Integer (upper bound): The rule matches any port with a number smaller than the integer.
  • Range integers: The rule matches any port whose number is between the integers.
• **Flag bits**: Rules filter TCP packets on flag bits.
• **Message type**: Rules filter ICMP type or code.
• **Tracked**: Matches packets in existing ICMP, UDP, or TCP connections. Valid in ACLs applied to the control plane. Validity in ACLs applied to the data plane varies by switch platform.

**Standard IPv4 and IPv6 ACL Rule Parameters**

<table>
<thead>
<tr>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>When calculating the size of ACLs, be aware that Arista switches install four rules in every IPv6 ACL so that ICMPv6 neighbor discovery packets bypass the default drop rule.</td>
</tr>
</tbody>
</table>

Standard ACLs filter only on the source address.

**MAC ACL Rule Parameters**

MAC ACLs filter traffic on a packet’s layer 2 header. Criteria that MAC ACLs use to filter packets include:

• **Source Address and Mask**: The packet’s source MAC address. Valid rule inputs include:
  • MAC address range (address-mask in 3x4 dotted hexadecimal notation).
  • any to denote that the rule matches all source addresses.
• **Destination Address and Mask**: The packet’s destination MAC address. Valid rule inputs include:
  • MAC address range (address-mask in 3x4 dotted hexadecimal notation).
  • any to denote that the rule matches all destination addresses.
• **Protocol**: The packet’s protocol as specified by its EtherType field contents. Valid inputs include:
  • Protocol name for a limited set of common protocols.
  • Assigned protocol number for all protocols.
24.2.1.3 Creating and Modifying Lists

The switch provides configuration modes for creating and modifying ACLs. The command that enters an ACL configuration mode specifies the name of the list that the mode modifies. The switch saves the list to the running configuration when the configuration mode is exited.

- ACLs are created and modified in ACL configuration mode.
- Standard ACLs are created and modified in Standard-ACL-configuration mode.
- MAC ACLs are created and modified in MAC-ACL-configuration mode.

Lists that are created in one mode cannot be modified in any other mode.

A sequence number designates the rule’s placement in a list. New rules are inserted into a list according to their sequence numbers. A rule’s sequence number can be referenced when deleting it from a list.

Section 24.2.2 describes procedures for configuring ACLs.

24.2.1.4 Implementing Access Control Lists

An access control list (ACL) is implemented by assigning the list to an Ethernet interface or subinterface, to a port channel interface, or to the control plane. The switch assigns a default ACL to the control plane unless the configuration contains a valid control-plane ACL assignment statement. Ethernet and port channel interfaces are not assigned an ACL by default. Standard ACLs are applied to interfaces in the same manner as other ACLs.

IPv4 and MAC ACLs are separately applied for inbound and outbound packets. An interface or subinterface can be assigned multiple ACLs, with a limit of one ACL per packet direction per ACL type. Egress ACLs are supported on a subset of all available switches. The control-plane does not support egress ACLs.

Section 24.2.3 describes procedures for applying ACLs to interfaces or the control plane.

24.2.1.5 ACL Rule Tracking

ACL rule tracking determines the impact of ACL rules on the traffic accessing interfaces upon which they are applied. ACLs provide two tracking mechanisms:

- ACL logging: A syslog entry is logged when a packet matches specified ACL rules.
- ACL counters: ACL counters increment when a packet matches a rule in specified ACLs.

**ACL Logging**

ACL rules provide a `log` option that produces a log message when a packet matches the rule. ACL logging creates a syslog entry when a packet matches an ACL rule where logging is enabled. Packets that match a logging-enabled ACL rule are copied to the CPU by the hardware. These packets trigger the creation of a syslog entry. The information provided in the entry depends on the ACL type or the protocol specified by the ACL. Hardware rate limiting is applied to packets written to the CPU, avoiding potential DoS attacks. The rate of logging is also software limited to avoid the creation of syslog lists that are too large for practical use by human operators.

Section 24.2.2.3 describes procedures for configuring and enabling ACL logging.
**ACL Counters**

An ACL counter is assigned to each ACL rule. The activity of the ACL counters for rules within a list depend on the list’s counter state. When the list is in counting state, the ACL counter of a rule increments when the rule matches a packet. When the list is in non-counting state, the counter does not increment. A list’s counter state applies to all rules in the ACL. The initial state for new ACLs is non-counting.

When an ACL changes from counting state to non-counting state, or when the ACL is no longer applied to any interfaces that increment counters, counters for all rules in the list maintain their values and do not reset. When the ACL returns to counting mode or is applied to an interface that increments counters, the counter operation resumes from its most recent value.

Counters never decrement and are reset only through CLI commands.

Section 24.2.2.3 describes procedures for configuring and enabling ACL counters.

### 24.2.1.6 AlgoMatch

AlgoMatch enables more flexible and scalable solutions for access control, telemetry, and enforcement networking whether the requirements are an on-premises or hybrid cloud model. By combining power-efficient and low-cost general purpose memory technology with advanced software algorithms, AlgoMatch provides greater scale, performance, and efficiency compared to common standard implementations with merchant silicon systems and TCAM.

In a typical TCAM solution, as additional lookup capacity is added, the power increases in-line with the scale. AlgoMatch utilizes power-efficient searching, only checking locations needed, and as a result lowers power draw by as much as half compared to TCAMs.

AlgoMatch utilizes a flexible and efficient packet matching algorithm with variable lookup sizes, rather than a fixed size lookup with TCAM. This enables full flow matching against source and destination criteria, or parts of the mask, and allows for multiple actions to be performed on a single packet or flow, with user defined filters for packet classification and custom actions in a single pass. A policy with multiple actions is not typically possible with a TCAM solution without using recirculation of chained lookups. However, with AlgoMatch, multiple actions can be applied in a single pass (For example, access control, telemetry, and counters) without losing either features or performance.

**AlgoMatch Benefits**

- Improved power efficiency: 50 percentage lower.
- Efficient Layer 4 rules: allows enhanced security policy.
- Counters: provide better visibility and flow filtering.

---

**Note**

AlgoMatch configuration and show commands are no different from security ACLs. For more information on ACL commands refer Section 24.7: ACL, Route Map, and Prefix List Commands.

### 24.2.2 ACL Configuration

Access control lists are created and modified in an ACL-configuration mode. A list can be edited only in the mode where it was created. The switch provides five configuration modes for creating and modifying access control lists:

- **ACL configuration mode** for IPv4 access control lists.
- **IPv6-ACL configuration mode** for IPv6 access control lists.
- **Std-ACL configuration mode** for Standard IPv4 access control lists.
Chapter 24: ACLs and Route Maps

Access Control Lists

- **Std-IPv6-ACL configuration mode** for Standard IPv6 access control lists.
- **MAC-ACL configuration mode** for MAC access control lists.

These sections describe the creation and modification of ACLs:

- Section 24.2.2.1: Managing ACLs
- Section 24.2.2.2: Modifying an ACL
- Section 24.2.2.3: ACL Rule Tracking Configuration
- Section 24.2.2.4: Displaying ACLs
- Section 24.2.2.5: Configuring Per-Port Per-VLAN QoS
- Section 24.2.2.6: Displaying Per-Port Per-VLAN QoS

### 24.2.2.1 Managing ACLs

#### Creating and Opening a List

To create an ACL, enter one of the following commands, followed by the name of the list:

- `ip access-list` for IPv4 ACLs.
- `ipv6 access-list` for IPv6 ACLs.
- `ip access-list standard` for standard IPv4 ACLs.
- `ipv6 access-list standard` for standard IPv6 ACLs.
- `mac access-list` for MAC ACLs.

The switch enters the appropriate ACL configuration mode for the list. If the command is followed by the name of an existing ACL, subsequent commands edit that list (see Modifying an ACL for additional information).

#### Examples

- This command places the switch in ACL configuration mode to create an ACL named **test1**.
  
  ```
  switch(config)#ip access-list test1
  switch(config-acl-test1)#
  ```

- This command places the switch in Standard-ACL-configuration mode to create a Standard ACL named **stest1**.
  
  ```
  switch(config)#ip access-list standard stest1
  switch(config-std-acl-stest1)#
  ```

- This command places the switch in MAC-ACL configuration mode to create an MAC ACL named **mtest1**.
  
  ```
  switch(config)#mac access-list mtest1
  switch(config-mac-acl-mtest1)#
  ```

#### Saving List Modifications

ACL configuration modes are group-change modes. Changes made in a group-change mode are saved by exiting the mode.

**Important!** After exiting ACL mode, the running-config file must be saved to the startup configuration file to preserve an ACL after a system restart.
**Example**

- The second example in *Adding a Rule* (page 1297) results in this edited ACL:

  ```text
  switch(config-acl-test1)#show
  IP Access List test1
     10 permit ip 10.10.10.0/24 any
     20 permit ip 10.30.10.0/24 host 10.20.10.1
     30 deny ip host 10.10.10.1 host 10.20.10.1
     40 permit ip any any
  ```

  Because the changes were not yet saved, the ACL remains empty, as shown by *show ip access-lists*.

  ```text
  switch(config-acl-test1)#show ip access-lists test1
  switch(config-acl-test1)#
  ```

  To save all current changes to the ACL and exit ACL configuration mode, type `exit`.

  ```text
  switch(config-acl-test1)#exit
  switch(config)#show ip access-lists test1
  IP Access List test1
     10 permit ip 10.10.10.0/24 any
     20 permit ip 10.30.10.0/24 host 10.20.10.1
     30 deny ip host 10.10.10.1 host 10.20.10.1
     40 permit ip any any
  ```

  **Discarding List Changes**

  The `abort` command exits ACL configuration mode without saving pending changes.

  **Example**

  - Example 2 in *Adding a Rule* (page 1297) results in this edited ACL:

    ```text
    switch(config-acl-test1)#show
    IP Access List test1
       10 permit ip 10.10.10.0/24 any
       20 permit ip 10.30.10.0/24 host 10.20.10.1
       30 deny ip host 10.10.10.1 host 10.20.10.1
       40 permit ip any any
    ```

    To discard the changes, enter `abort`. If the ACL existed before entering ACL-configuration mode, `abort` restores the version that existed before entering ACL-configuration mode. Otherwise, *show ip access-lists* shows the ACL was not created.

    ```text
    switch(config-acl-test1)#abort
    switch(config)#
    ```

**24.2.2.2 Modifying an ACL**

An existing ACL, including those currently applied to interfaces, can be modified by entering the appropriate configuration mode for the ACL as described in *Creating and Opening a List*. By default, while an ACL is being modified all traffic is blocked on any interface to which the ACL has been applied.

**Permit All Traffic During ACL Update**

Because blocking ports during ACL modifications can result in packet loss and can interfere with features such as routing and dynamic NAT, 7050X, 7060X, 7150, 7250X, 7280, 7280R, 7300X, 7320X, and 7500 series switches can be configured instead to permit all traffic on Ethernet and VLAN interfaces while ACLs applied to those interfaces are being modified. This is done with the `hardware access-list update default-result permit` command.
These commands add deny rules to the appropriate ACL:

- `deny (IPv4 ACL)` adds a deny rule to an IPv4 ACL.
- `deny (IPv6 ACL)` adds a deny rule to an IPv6 ACL.
- `deny (Standard IPv4 ACL)` adds a deny rule to an IPv4 standard ACL.
- `deny (Standard IPv6 ACL)` adds a deny rule to an IPv6 standard ACL.
- `deny (MAC ACL)` adds a deny rule to a MAC ACL.

These commands add permit rules to the appropriate ACL:

- `permit (IPv4 ACL)` adds a permit rule to an IPv4 ACL.
- `permit (IPv6 ACL)` adds a permit rule to an IPv6 ACL.
- `permit (Standard IPv4 ACL)` adds a permit rule to an IPv4 standard ACL.
- `permit (Standard IPv6 ACL)` adds a permit rule to an IPv6 standard ACL.
- `permit (MAC ACL)` adds a permit rule to a MAC ACL.

### Adding a Rule

To append a rule to the end of a list, enter the rule without a sequence number while in ACL configuration mode for the list. The new rule’s sequence number is derived by adding 10 to the last rule’s sequence number.

### Examples

- This command configures the switch to permit all traffic during ACL modifications on interfaces to which the ACL has been applied. The rules in modified ACLs are applied after exiting ACL configuration mode, and after the ACL rules have been populated in hardware.

  ```
  switch(config)#hardware access-list update default-result permit
  
  switch(config-acl-test1)#permit ip 10.10.10.0/24 any
  switch(config-acl-test1)#permit ip any host 10.20.10.1
  switch(config-acl-test1)#deny ip host 10.10.10.1 host 10.20.10.1
  
  To view the edited list, type `show`.
  ```

  ```
  switch(config-acl-test1)#show
  IP Access List test1
  10 permit ip 10.10.10.0/24 any
  20 permit ip any host 10.20.10.1
  30 deny ip host 10.10.10.1 host 10.20.10.1
  
  switch(config-acl-test1)#permit ip any any
  
  switch(config-acl-test1)#show
  IP Access List test1
  10 permit ip 10.10.10.0/24 any
  20 permit ip any host 10.20.10.1
  30 deny ip host 10.10.10.1 host 10.20.10.1
  40 permit ip any any
  ```

- This command appends a rule to the ACL. The new rule’s sequence number is 40.

### Inserting a Rule

To insert a rule into a ACL, enter the rule with a sequence number between the existing rules’ numbers.
Example

- This command inserts a rule between the first two rules by assigning it the sequence number 15.

```
Switch(config-acl-test1)#15 permit ip 10.30.10.0/24 host 10.20.10.1
Switch(config-acl-test1)#show
IP Access List test1
  10 permit ip 10.10.10.0/24 any
  15 permit ip 10.30.10.0/24 host 10.20.10.1
  20 permit ip any host 10.20.10.1
  30 deny ip host 10.10.10.1 host 10.20.10.1
  40 permit ip any any
```

Deleting a Rule

To remove a rule from the current ACL, perform one of these commands:

- Enter `no`, followed by the sequence number of the rule to be deleted.
- Enter `no`, followed by the rule be deleted.
- Enter `default`, followed by the rule to be deleted.

Example

- These equivalent commands remove rule 20 from the list.

```
switch(config-acl-test1)#no 20

switch(config-acl-test1)#no permit ip any host 10.20.10.1

switch(config-acl-test1)#default permit ip any host 10.20.10.1
```

This ACL results from entering one of the preceding commands.

```
switch(config-acl-test1)#show
ip access list test1
  10 permit ip 10.10.10.0/24 any
  15 permit ip 10.30.10.0/24 host 10.20.10.1
  30 deny ip host 10.10.10.1 host 10.20.10.1
  40 permit ip any any
```

Resequencing Rule Numbers

Sequence numbers determine the order of the rules in an access control list. After a list editing session where existing rules are deleted and new rules are inserted between existing rules, the sequence number distribution may not be uniform. Resequencing rule numbers changes the sequence number of rules to provide a constant difference between adjacent rules. The `resequence (ACLs)` command adjusts the sequence numbers of ACL rules.
Example

- The `resequence` command renames rules in the test1 ACL. The sequence number of the first rule is 100; subsequent rule numbers are incremented by 20.

  ```
  switch(config-acl-test1)#show
  IP Access List test1
  10 permit ip 10.10.10.0/24 any
  25 permit ip any host 10.20.10.1
  30 deny ip host 10.10.10.1 host 10.20.10.1
  50 permit ip any any
  90 remark end of list
  switch(config-acl-test1)#resequence 100 20
  switch(config-acl-test1)#show
  IP Access List test1
  100 permit ip 10.10.10.0/24 any
  120 permit ip any host 10.20.10.1
  140 deny ip host 10.10.10.1 host 10.20.10.1
  160 permit ip any any
  180 remark end of list
  ```

24.2.2.3 ACL Rule Tracking Configuration

ACL rules provide a log option that produces a syslog message about the packets matching packet. ACL logging creates a syslog entry when a packet matches an ACL rule with logging enabled.

This feature is currently available on Arad switches and on 7100 series switches. On 7100 series switches, matches are logged only on ingress, not on egress.

Example

- This command creates an ACL rule with logging enabled.

  ```
  switch(config-acl-test1)#15 permit ip 10.30.10.0/24 host 10.20.10.1 log
  switch(config-acl-test1)#
  ```

The format of the generated syslog message depends on the ACL type and the specified protocol:

- Messages generated by a TCP or UDP packet matching an IP ACL use this format:
  
  **IPACCESS: list acl intf filter protocol src-ip(src-port) -> dst-ip(dst_port)**

- Messages generated by ICMP packets matching an IP ACL use this format:
  
  **IPACCESS: list acl intf filter icmp src-ip(src-port) -> dst-ip(dst-port) type=n code=m**

- Messages generated by all other IP packets matching an IP ACL use this format:
  
  **IPACCESS: list acl intf filter protocol src-ip -> dst-ip**

- Messages generated by packets matching a MAC ACL use this format:
  
  **MACACCESS: list acl intf filter vlan ether src_mac -> dst_mac**

- Messages generated by a TCP or UDP packet matching a MAC ACL use this format:
  
  **MACACCESS: list acl intf filter vlan ether ip-prt src-mac src-ip: src-ptr -> dst-mac dst-ip: dst-ptr**

- Messages generated by any other IP packet matching a MAC ACL use this format:
  
  **MACACCESS: list acl intf filter vlan ether src_mac src_ip -> dst_mac dst_ip**

Variables in the syslog messages display the following values:

- `acl` Name of ACL.
Access Control Lists

Chapter 24: ACLs and Route Maps

- **intf** Name of interface that received the packet.
- **filter** Action triggered by ACL (denied or permitted).
- **protocol** IP protocol specified by packet.
- **vlan** Number of VLAN receiving packet.
- **ether** EtherType protocol specified by packet.
- **src-ip** and **dst-ip** source and destination IP addresses.
- **src-prt** and **dst-prt** source and destination ports.
- **src-mac** and **dst-mac** source and destination MAC addresses.

ACLs provide a command that configures its counter state (counting or non-counting). The counter state applies to all rules in the ACL. The initial state for new ACLs is non-counting.

The **counters per-entry (ACL configuration modes)** command places the ACL in counting mode.

- This command places the configuration mode ACL in counting mode.

```plaintext
switch(config-acl-test1)#counters per-entry
switch(config-acl-test1)#exit
switch(config-acl-test1)#show ip access-list test1
```

```
IP Access List test1
    counters per-entry
    10 permit ip 10.10.10.0/24 any
    20 permit ip any host 10.20.10.1
    30 deny ip host 10.10.10.1 host 10.20.10.1
    40 permit ip any any
    50 remark end of list
```

The **clear ip access-lists counters** and **clear ipv6 access-lists counters** commands set the IP access list counters to zero for the specified IP access list.

- This command clears the ACL counter for the test1 ACL.

```plaintext
switch(config)#clear ip access-lists counters test1
switch(config)#
```

### 24.2.2.4 Displaying ACLs

ACLs can be displayed by a **show running-config** command. The **show ip access-lists** also displays ACL rosters and contents, as specified by command parameters.

When editing an ACL, the **show (ACL configuration modes)** command displays the current or pending list, as specified by command parameters.

**Displaying a List of ACLs**

To display the roster of ACLs on the switch, enter **show ip access-lists** with the **summary** option.
Example

- This command lists the available access control lists.

switch(config)#show ip access-list summary
IPV4 ACL default-control-plane-acl
    Total rules configured: 12
    Configured on: control-plane
    Active on : control-plane

IPV4 ACL list2
    Total rules configured: 3

IPV4 ACL test1
    Total rules configured: 6

IPV4 ACL test_1
    Total rules configured: 1

IPV4 ACL test_3
    Total rules configured: 0

switch(config)#

Displaying Contents of an ACL

These commands display ACL contents.

- show ip access-lists
- show ipv6 access-lists
- show mac access-lists

Each command can display the contents of one ACL or of all ACLs of the type specified by the command:

- To display the contents of one ACL, enter show ip access-lists followed by the name of the ACL.
- To display the contents of all ACLs on the switch, enter the command without any options.

ACLs that are in counting mode display the number of inbound packets each rule in the list matched and the elapsed time since the last match.

Example

- This command displays the rules in the default-control-plane-acl ACL.

switch#show ip access-lists default-control-plane-acl
IP Access List default-control-plane-acl [readonly]
    counters per-entry
    10 permit icmp any any
    20 permit ip any any tracked [match 1725, 0:00:00 ago]
    30 permit ospf any any
    40 permit tcp any any eq ssh telnet www snmp bgp https
    50 permit udp any any eq bootps bootpc snmp [match 993, 0:00:29 ago]
    60 permit tcp any any eq mlag ttl eq 255
    70 permit udp any any eq mlag ttl eq 255
    80 permit vrrp any any
    90 permit ahp any any [match 1316, 0:00:23 ago]
    100 permit pim any any
    110 permit igmp any any [match 1316, 0:00:23 ago]
    120 permit tcp any any range 5900 5910
- This command displays the rules in all ACLs on the switch.

```
switch#show ip access-lists
IP Access List default-control-plane-acl [readonly]
  counters per-entry
  10 permit icmp any any
  20 permit ip any any tracked [match 1371, 0:00:00 ago]
  30 permit ospf any any
  40 permit tcp any any eq ssh telnet www snmp bgp https
  50 permit udp any any eq bootps bootpc snmp
  60 permit tcp any any eq mlag ttl eq 255
  70 permit udp any any eq mlag ttl eq 255
  80 permit vrrp any any
  90 permit ahp any any
  100 permit pim any any
  110 permit igmp any any [match 1316, 0:00:23 ago]
  120 permit tcp any any range 5900 5910

IP Access List list2
  10 permit ip 10.10.10.0/24 any
  20 permit ip 10.30.10.0/24 host 10.20.10.1
  30 permit ip any host 10.20.10.1
  40 deny ip host 10.10.10.1 host 10.20.10.1
  50 permit ip any any

IP Access List test1

Switch(config)#
```

### Displaying ACL Modifications

While editing an ACL in ACL-configuration mode, the `show (ACL configuration modes)` command provides options for displaying ACL contents.

- To display the list, as modified in ACL configuration mode, enter `show` or `show pending`.
- To display the list, as stored in `running-config`, enter `show active`.
- To display differences between the pending list and the stored list, enter `show diff`.

### Examples

The examples in this section assume these ACL commands were previously entered.

**These commands are stored in the configuration:**

```
10 permit ip 10.10.10.0/24 any
20 permit ip any host 10.21.10.1
30 deny ip host 10.10.10.1 host 10.20.10.1
40 permit ip any any
50 remark end of list
```

**The current edit session removed this command. This change is not yet stored to running-config:**

```
20 permit ip any host 10.21.10.1
```
The current edit session added these commands ACL. They are not yet stored to running-config:

- 20 permit ip 10.10.0.0/16 any
- 25 permit tcp 10.10.20.0/24 any
- 45 deny pim 239.24.124.0/24 10.5.8.4/30

- This command displays the pending ACL, as modified in ACL configuration mode.

```
switch(config-acl-test_1)#show pending
IP Access List test_1
  10 permit ip 10.10.10.0/24 any
  20 permit ip 10.10.0.0/16 any
  25 permit tcp 10.10.20.0/24 any
  30 deny ip host 10.10.10.1 host 10.20.10.1
  40 permit ip any any
  45 deny pim 239.24.124.0/24 10.5.8.4/30
  50 remark end of list
```

- This command displays the ACL, as stored in the configuration.

```
switch(config-acl-test_1)#show active
IP Access List test_1
  10 permit ip 10.10.10.0/24 any
  20 permit ip any host 10.21.10.1
  30 deny ip host 10.10.10.1 host 10.20.10.1
  40 permit ip any any
  50 remark end of list
```

- This command displays the difference between the saved and modified ACLs.

```
 switch(config-acl-test_1)#show diff
---
+++ 
@@ -1,7 +1,9 @@
 IP Access List test_1
-  10 permit ip 10.10.10.0/24 any
-  20 permit ip any host 10.21.10.1
-  30 deny ip host 10.10.10.1 host 10.20.10.1
-  40 permit ip any any
-  50 remark end of list
+  10 permit ip 10.10.10.0/24 any
+  20 permit ip 10.10.0.0/16 any
+  25 permit tcp 10.10.20.0/24 any
+  30 deny ip host 10.10.10.1 host 10.20.10.1
+  40 permit ip any any
+  45 deny pim 239.24.124.0/24 10.5.8.4/30
```

24.2.2.5 Configuring Per-Port Per-VLAN QoS

To configure per-port per-VLAN QoS, first, configure the ACL policing for QoS, and then apply the policy-map on a single Ethernet or port-channel interfaces on a per-port per-VLAN basis. The per port per VLAN QoS allows a class-map to match traffic for a single VLAN or for a range of VLANs separated by commas.

**Note**

To configure per-port per-VLAN QoS on DCS-7280(E/R) and DCS-7500(E/R), change the **TCAM profile** to **QoS** as shown below.

**Step 1** Change the TCAM profile to QoS.

```
switch#config
switch(config)#
switch(config)#hardware tcam profile qos
```
Step 2  Create an ACL and then match the traffic packets based on the VLAN value and the VLAN
mask configured in the ACL.

```
switch(config)#ip access-list acl1
switch(config-acl-acl1)#permit vlan 100 0xfff ip any any
switch(config-acl-acl1)#exit
```

Step 3  Similarly, create a class-map and then match the traffic packets based on the range of VLAN
values configured in the class-map.

```
switch(config)#class-map match-any class1
switch(config-cmap-qos-class1)#match vlan 20-40, 1000-1250, 2000
switch(config-cmap-qos-class1)#exit
```

Note  In ACLs, the VLAN configuration must have a VLAN mask, whereas the class-map allows the VLAN
configuration without a VLAN mask, in such cases use 0xFFF as the default VLAN mask.

24.2.2.6 Displaying Per-Port Per-VLAN QoS

The following show commands display the status, traffic hit counts, tcam profile information, and
policy-maps configured on an interface.

- The `show policy-map` command displays the policy-map information of the configured
  policy-map.

Example

```
switch#show policy-map policy1
Service-policy policy1
Class-map: class1 (match-any)
Match: ip access-group name acl1
Police cir 512000 bps bc 96000 bytes
Class-map: class-default (match-any)
```

- The `show policy-map interface` command displays the policy-map configured on an interface.

Example

```
switch#show policy-map interface ethernet 1
Service-policy input: p1
Hardware programming status: Successful
Class-map: c2001 (match-any)
Match: vlan 2001 0xfff
set dscp 4
Class-map: c2002 (match-any)
Match: vlan 2002 0xfff
set dscp 8
Class-map: c2003 (match-any)
Match: vlan 2003 0xfff
set dscp 12
```

24.2.2.7 On DCS-7010, DCS-7050X, DCS7250X, DCS-7300X series switches

- The `show policy-map policy-name counters` command displays the policy-map traffic match
count for the policy-map configured.
Example

```
switch#show policy-map policy1 counters
| Service-policy input: policy1 |
| Hardware programming status: Successful |
| Class-map: class1 (match-any) |
| Match: vlan 20-40,1000-1250 |
| police rate 100 mbps burst-size 100 kbytes |
| Interface: Ethernet16/1 |
| Conformed 28621 packets, 7098008 bytes -------------- packet match count |

Class-map: class-default (match-any)
Matched Packets: 19 -------------- packet match count
```

- The `show platform trident tcam [detail]` displays the TCAM entries configured for each TCAM group including policy-maps and corresponding hits.

Example

```
switch#show platform trident tcam

--- TCAM summary for switch Linecard0/0 ---
TCAM group 9 uses 42 entries and can use up to 1238 more.
  Mlag control traffic uses 4 entries.
  CVX traffic uses 6 entries.
  L3 Control Priority uses 23 entries.
  IGMP Snooping Flooding uses 8 entries.
  L4 MicroBfd traffic uses 1 entries.
TCAM group 13 uses 99 entries and can use up to 1181 more.
  Dot1x MAB traffic uses 1 entries.
  ACL Management uses 10 entries.
  Vxlan Traffic uses 24 entries.
  L2 Control Priority uses 11 entries.
  Storm Control Management uses 2 entries.
  ARP Inspection uses 2 entries.
  L3 Routing uses 49 entries.
TCAM group 14 uses 12 entries and can use up to 2548 more.
  Policy QOS uses 12 entries.
TCAM group 16 uses 59 entries and can use up to 1221 more.
  PDP Reserved uses 1 entries.
  PDP uses 58 entries.
TCAM group 67 uses 12 entries and can use up to 500 more.
  PDP Class Reservation uses 12 entries.
```
Example

```
switch#show platform trident tcam detail
=== TCAM detail for switch Linecard0/0 ===
TCAM group 9 uses 42 entries and can use up to 1238 more.
Mlag control traffic uses 4 entries.
  589826  0 hits - MLAG - SrcPort UDP Entry
  589827  0 hits - MLAG - DstPort UDP Entry
  589828  0 hits - MLAG - SrcPort TCP Entry
  589829  0 hits - MLAG - DstPort TCP Entry
CVX traffic reserves 6 entries (0 used).
L3 Control Priority uses 23 entries.
  589836  0 hits - URM - SelfIp UDP Entry
  589837  0 hits - URM - SelfIp TCP Entry
  589838  0 hits - URM - Ttl1 UDP Entry
  589839  0 hits - URM - Ttl1 TCP Entry
  589840  0 hits - BGP - Dst Port
  589841  0 hits - BGP - Src Port
  589842  0 hits - VRRP
  589843  0 hits - BFD
  589844  0 hits - BFD Multihop
  589845  0 hits - PIM
  589846  0 hits - PIM Null Register
  589847  0 hits - PIM Register
  589848  0 hits - OSPF - unicast
  589849  71196 hits - OSPFv2 - Multicast
  589850  0 hits - OSPFv3 - Multicast
  589851  0 hits - OSPF Auth ESP - Multicast
  589852  0 hits - OSPF Auth ESP - Unicast
  589853  0 hits - IP packets with GRE type and ISIS protocol
  589854  0 hits - RouterL3 Vlan Priority 6,7 Elevator
  589855  0 hits - RouterL3 DSCP 48-63 Elevator
  589856  0 hits - RouterL3 Priority Elevator
  589857  0 hits - NextHopToCpu, Clean
  589858  0 hits - L3MC Cpu OIF
IGMP Snooping Flooding reserves 8 entries (6 used).
  589860  0 hits - Drop All IGMP packets
  589861  0 hits - Flood link local packets
  589862  0 hits - IGMP Snooping Restricted Flooding L2 from local
              mlag peer
  589863  0 hits - IGMP Snooping Restricted Flooding L2
  589864  0 hits - IGMP Snooping Restricted Flooding L3 from local
              mlag peer
  589865  0 hits - IGMP Snooping Restricted Flooding L3
L4 MicroBfd traffic reserves 1 entries (0 used).
TCAM group 13 uses 99 entries and can use up to 1181 more.
Dot1x MAB traffic uses 1 entries.
  851968  0 hits - Dot1xMab Rule
```

<--------OUTPUT OMITTED FROM EXAMPLE-------->

```
ck338.22:14:38(config-pmap-qos-policy1)#
```

24.2.2.8 On DCS-7280(E/R), DCS7500(E/R) series switches

- The `show platform fap [fapName] acl tcam hw` command displays the TCAM entries configured for each TCAM bank including policy-maps and corresponding traffic match.
## Example

```bash
switch#show platform fap Arad1 acl tcam hw
```

---

Arad1 Bank 0 Type: dbPdpIp, dbPdpIp6, dbPdpMpls, dbPdpNonIp, dbPdpTunnel

---

```
<table>
<thead>
<tr>
<th>Offs</th>
<th>X</th>
<th>PR</th>
<th>TT</th>
<th>R</th>
<th>QI</th>
<th>V6MC</th>
<th>DPRT</th>
<th>SPRT</th>
<th>F</th>
<th>DEST</th>
<th>V</th>
<th>ACT</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>4</td>
<td>59</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>31</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>32</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>33</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>34</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>35</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>36</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>37</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>38</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>39</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>40</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>41</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>42</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>43</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>44</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>45</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>46</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>47</td>
<td>4</td>
<td>53</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
</tbody>
</table>
```

---

```
<table>
<thead>
<tr>
<th>Offs</th>
<th>X</th>
<th>LEM2</th>
<th>LIFP</th>
<th>QI</th>
<th>V6MC</th>
<th>DPRT</th>
<th>SPRT</th>
<th>F</th>
<th>DEST</th>
<th>V</th>
<th>ACT</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>63</td>
<td>6</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>000cb</td>
<td>0</td>
</tr>
<tr>
<td>64</td>
<td>6</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>000cb</td>
<td>0</td>
</tr>
<tr>
<td>65</td>
<td>6</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>000cb</td>
<td>0</td>
</tr>
<tr>
<td>66</td>
<td>6</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>000cb</td>
<td>0</td>
</tr>
<tr>
<td>67</td>
<td>6</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>000cb</td>
<td>0</td>
</tr>
<tr>
<td>68</td>
<td>6</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>000cb</td>
<td>0</td>
</tr>
<tr>
<td>69</td>
<td>6</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>000cb</td>
<td>0</td>
</tr>
<tr>
<td>70</td>
<td>6</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>000cb</td>
<td>0</td>
</tr>
<tr>
<td>71</td>
<td>6</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>000cb</td>
<td>0</td>
</tr>
</tbody>
</table>
```
<table>
<thead>
<tr>
<th>Offs</th>
<th>X</th>
<th>TT0</th>
<th>P6</th>
<th>SPRT</th>
<th>R</th>
<th>QI</th>
<th>PN</th>
<th>F</th>
<th>MC</th>
<th>O</th>
<th>DPRT</th>
<th>DEST</th>
<th>V4_DIP</th>
<th>V</th>
<th>ACT</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>02</td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>26ffd</td>
<td></td>
<td>3</td>
<td>00087</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>02</td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>e00000d</td>
<td>3</td>
<td>0008b</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>59</td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>e00000d</td>
<td>0</td>
<td>00000</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>59</td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>e00000d</td>
<td>0</td>
<td>00000</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>e00000d</td>
<td>0</td>
<td>00000</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>64</td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>26ffd</td>
<td></td>
<td>3</td>
<td>00089</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>64</td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>26ffd</td>
<td></td>
<td>0</td>
<td>00000</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>ff</td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>00b3</td>
<td>26ffd</td>
<td>3</td>
<td>0009b</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>ff</td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>00b3</td>
<td>26ffd</td>
<td>0</td>
<td>00000</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>ff</td>
<td>1150</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>1150</td>
<td>26ffd</td>
<td>0</td>
<td>00000</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Access Control Lists

Chapter 24: ACLs and Route Maps
24.2.3 Applying ACLs

Access control lists become active when they are assigned to an interface or subinterface or to the control plane. This section describes the process of adding and removing ACL interface assignments.

Applying an ACL to an Interface

The switch must be in interface configuration mode to assign an ACL to an interface or subinterface.
• The `ip access-group` command applies the specified IP or standard IP ACL to the configuration mode interface or subinterface.

• The `mac access-group` command applies the specified MAC ACL to the configuration mode interface.

IPv4, IPv6, and MAC ACLs are separately applied for inbound and outbound packets. An interface or subinterface can be assigned with multiple ACLs, with a limit of one ACL per packet direction per ACL type. Egress ACLs are supported on a subset of all available switches. IPv6 egress ACLs have limited availability, and IPv6 egress ACLs applied to routed interfaces or subinterfaces across the same chip on the DCS-7500E and the DCS-7280E series can be shared. In addition to that, the DSCP value can match on IPv6 egress ACLs. This result in a more efficient utilization of system resources, and is particularly useful for environments with few, potentially large, IPv6 egress ACLs applied across multiple routed interfaces.

**Example**

• These commands assign `test1` ACL to Ethernet interface 3, then verify the assignment.

  ```
  switch(config)#interface ethernet 3
  switch(config-if-Et3)#ip access-group test1 in
  switch(config-if-Et3)#show running-config interfaces ethernet 3
  interface Ethernet3
    ip access-group test1 in
  switch(config-if-Et3)#
  ```

• This command enables shared ACLs.

  ```
  switch(config)#hardware access-list resource sharing vlan ipv6 out
  switch(config)#
  ```

• This command disables shared ACLs.

  ```
  switch(config)#no hardware access-list resource sharing vlan ipv6 out
  switch(config)#
  ```

• These commands apply an IPv4 ACL named “test_ACL” to ingress traffic on Ethernet subinterface 5.1.

  ```
  switch(config)#interface ethernet 5.1
  switch(config-if-Et5.1)#ipv4 access-group test_ACL in
  switch(config-if-Et5.1)#
  ```

**Removing an ACL from an Interface**

The `no ip access-group` command removes an IP ACL assignment statement from `running-config` for the configuration mode interface. After an ACL is removed, the interface is not associated with an IP ACL.

The `no mac ip access-group` command removes a MAC ACL assignment statement from `running-config` for the configuration mode interface. After a MAC ACL is removed, the interface is not associated with an MAC ACL.

To remove an ACL from the control plane, enter the `no ip access-group` command in control plane configuration mode. Removing the control plane ACL command from `running-config` reinstates `default-control-plane-acl` as the control plane ACL.

**Examples**

• These commands remove the assigned IPv4 ACL from Ethernet interface 3.

  ```
  switch(config)#interface ethernet 3
  switch(config-if-Et3)#no ip access-group test in
  switch(config-if-Et3)#
  ```
• These commands place the switch in control plane configuration mode and remove the ACL assignment from `running-config`, restoring `default-control-plane-acl` as the Control Place ACL.

```
switch(config)#control-plane
switch(config-cp)#no ip access-group test_cp in
switch(config-cp)#
```
24.3 Service ACLs

These sections describe Service ACLs:

- Section 24.3.1: Service ACL Description
- Section 24.3.2: Configuring Service ACLs and Displaying Status and Counters

24.3.1 Service ACL Description

Service ACL enforcement is a feature added to a control plane service (the SSH server, the SNMP server, routing protocols, etc) that allows the switch administrator to restrict the processing of packets and connections by the control plane processes that implement that service. The control plane program run by the control plane process checks already received packets and connections against a user configurable access control list (ACL), a Service ACL. The Service ACL contains permit and deny rules matching any of the source address, destination address, and TCP or UDP ports of received packets or connections. After receiving a packet or connection, the control plane process evaluates the packet or connection against the rules of the Service ACL configured for the control plane process, and if the received packet or connection matches a deny rule the control plane process drops or closes it without further processing.

Control Plane Process Enforced Access Control enables the system administrator to restrict which systems on the network can access the services provided by the switch. Each service has its own access control list, giving the system administrator fine grained control over access to the switch's control plane services. The CLI for this uses the familiar pattern of access control lists assigned for a specific purpose, in this case for each control plane service.

24.3.2 Configuring Service ACLs and Displaying Status and Counters

24.3.2.1 SSH Server

To apply the SSH server Service ACLs for IPv4 and IPv6 traffic, use the `ip access-group (Service ACLs)` and `ipv6 access-group (Service ACLs)` commands in mgt-ssh configuration mode as shown below.

```
(config)# management ssh
(config-mgmt-ssh)# ip access-group <acl_name> [vrf <vrf_name>] in
(config-mgmt-ssh)# ipv6 access-group <acl_name> [vrf <vrf_name>] in
```

In EOS 4.19.0, all VRFs are required to use the same SSH server Service ACL. The Service ACL assigned without the `vrf` keyword is applied to all VRFs where the SSH server is enabled.

To display the status and counters of the SSH server Service ACLs, use the following commands.

```
(switch)# show management ssh ip access-list
(switch)# show management ssh ipv6 access-list
```

24.3.2.2 SNMP Server

To apply the SNMP server Service ACLs to restrict which hosts can access SNMP services on the switch, use the `snmp-server community` command as shown below.

```
(config)# snmp-server community <community-name> [view <viewname>] [ro | rw] <acl_name>
(config)# snmp-server community <community-name> [view <viewname>] [ro | rw] ipv6 <ipv6_acl_name>
```
24.3.2.3 **EAPI**

To apply Service ACLs to the EOS application programming interface (EAPI) server, use the `ip access-group` (Service ACLs) and `ipv6 access-group` (Service ACLs) commands as shown below.

```
(config)# management api http-commands
(config-mgmt-api-http-cmds)# vrf <vrf_name>
(config-mgmt-api-http-cmds-vrf-<vrf>)# ip access-group <acl_name>
(config-mgmt-api-http-cmds-vrf-<vrf>)# ipv6 access-group <ipv6_acl_name>
```

**Note**

To configure a Service ACL for the EAPI server in the default VRF, use the `vrf default` command to enter the per-VRF configuration mode for the default VRF before using the `ip access-group` (Service ACLs) or `ipv6 access-group` (Service ACLs) command.

To display the status and counters of the EAPI server Service ACLs, use the following commands.

```
(switch)> show management api http-commands ip access-list
(switch)> show management api http-commands ipv6 access-list
```

24.3.2.4 **BGP**

To apply Service ACLs for controlling connections to the BGP routing protocol agent, use the `ip access-group` (Service ACLs) and `ipv6 access-group` (Service ACLs) commands as shown below.

```
(config)# router bgp <asn>
(config-router-bgp)# ip access-group <acl_name>
(config-router-bgp)# ipv6 access-group <ipv6_acl_name>
(config-router-bgp)# vrf <vrf_name>
(config-router-bgp-vrf-<vrf>)# ip access-group <acl_name>
(config-router-bgp-vrf-<vrf>)# ipv6 access-group <ipv6_acl_name>
```

To display the status and counters of the BGP routing protocol Service ACLs, use the following commands.

```
(switch)> show bgp ipv4 access-list
(switch)> show bgp ipv6 access-list
```

24.3.2.5 **OSPF**

To apply Service ACLs for controlling packets processed by the OSPF routing protocol agent, use the `ip access-group` (Service ACLs) and `ipv6 access-group` (Service ACLs) commands as shown below.

```
(config)# router ospf <id>
(config-router-ospf)# ip access-group <acl_name>
(config-router-ospf)# ipv6 access-group <ipv6_acl_name>
```

When using VRFs, each per-VRF OSPF instance must be assigned its Service ACL explicitly.

To display the status and counters of the OSPF routing protocol Service ACLs, use the following commands.

```
(switch)> show ospf ipv4 access-list
(switch)> show ospf ipv6 access-list
```
24.3.2.6 PIM

To apply Service ACLs for controlling packets processed by the PIM routing protocol agent, use the `access-group` command as shown below.

```
(config)#router pim
(config-router-pim)#ipv4
(config-router-pim-ipv4)#access-group <acl_name>
(config-router-pim-ipv4)#vrf <vrf_name>
(config-router-pim-vrf-<vrf>)#ipv4
(config-router-pim-vrf-<vrf>-ipv4)#access-group <acl_name>
```

To display the status and counters of the PIM routing protocol Service ACLs, use the following commands.

```
(switch)> show ip pim access-list
```

24.3.2.7 IGMP

To apply Service ACLs for controlling packets processed by the IGMP management protocol agent, use the `ip igmp access-group` command as shown below.

```
(config)# router igmp
(config-router-igmp)# ip igmp access-group <acl_name>
(config-router-igmp)# vrf <vrf_name>
(config-router-igmp-vrf-<vrf>)# ip igmp access-group <acl_name>
```

To display the status and counters of the IGMP management protocol Service ACLs, use the following commands.

```
(switch)> show ip igmp access-list
```

24.3.2.8 DHCP Relay

To apply Service ACLs for controlling packets processed by the DHCP relay agent, use the `ip dhcp relay access-group` and `ipv6 dhcp relay access-group` commands as shown below.

```
(config)# ip dhcp relay access-group <acl_name> [vrf <vrf_name>]
(config)# ipv6 dhcp relay access-group <acl_name> [vrf <vrf_name>]
```

To display the status and counters of the DHCP relay agent Service ACLs, use the following commands.

```
(switch)> show ip dhcp relay access-list
(switch)> show ipv6 dhcp relay access-list
```

24.3.2.9 LDP

To apply Service ACLs for controlling packets and connections processed by the LDP MPLS label distribution protocol, use the `ip access-group (Service ACLs)` command as shown below.

```
(config)# mpls ldp
(config-mpls-ldp)# ip access-group <acl_name>
```

To display the status and counters of the LDP Service ACLs, use the following command.

```
(switch)> show mpls ldp access-list
```
24.3.2.10 LANZ
To apply Service ACLs for controlling connections accepted by the LANZ agent, use the ip access-group (Service ACLs) and ipv6 access-group (Service ACLs) commands as shown below.

```
(config)# queue-monitor streaming
(config-qm-streaming)# ip access-group <acl_name>
(config-qm-streaming)# ipv6 access-group <ipv6_acl_name>
```
To display the status and counters of the LDP Service ACLs, use the following command.

```
(switch)> show queue-monitor streaming access-lists
```

24.3.2.11 MPLS Ping and Traceroute
To apply Service ACLs for controlling connections accepted by the MPLS Ping agent, use the ip access-group (Service ACLs) and ipv6 access-group (Service ACLs) commands as shown below.

```
(config)# mpls ping
(config-mpls-ping)# ip access-group <acl_name> [vrf <vrf_name>]
(config-mpls-ping)# ipv6 access-group <ipv6_acl_name> [vrf <vrf_name>]
```

24.3.2.12 Telnet Server
To apply Service ACLs to the Telnet server, use the ip access-group (Service ACLs) and ipv6 access-group (Service ACLs) commands as shown below.

```
(config)# management telnet
(config-mgmt-telnet)# ip access-group <acl_name> [vrf <vrf_name>] in
(config-mgmt-telnet)# ipv6 access-group <ipv6_acl_name> [vrf <vrf_name>] in
```
In EOS 4.19.0, all VRFs are required to use the same Telnet server Service ACL. The Service ACL assigned without the vrf keyword is applied to all VRFs where the Telnet server is enabled.

To display the status and counters of the LDP Service ACLs, use the following commands.

```
(switch)>show management telnet ip access-list
(switch)>show management telnet ipv6 access-list
```
24.4 RACL Sharing on SVIs

24.4.1 IPv4 Ingress Sharing

The IPv4 ingress sharing optimizes the utilization of hardware resources by sharing the hardware resources between different VLAN interfaces when they have same ACL attached.

Larger deployments are benefited with this function, where IPv4 ingress sharing is applied on multiple SVIs with member interfaces on same forwarding ASIC. For example, a trunk port carrying multiple VLANs and an ingress sharing is applied on all VLANs, it occupies lesser hardware resources irrespective of number of VLANs. By default, IPv4 ingress sharing is disabled on the switches.

To enable IPv4 Ingress Sharing use [no] `hardware access-list resource sharing vlan in` command. Note, enabling or disabling the IPv4 ingress sharing requires the restart of software agents on the switches which is a disruptive process and will impact the traffic forwarding. The `no` form of the command disables the IPv4 ingress sharing on the switch. To display the IPv4 ingress sharing information use `show platform trident` command on the switch.

24.4.2 IPv4 Egress Sharing

The IPv4 Egress Sharing optimizes the utilization of hardware resources by sharing TCAM entries for a group of SVIs on which IPv4 ACLs shared. The TCAM entries are shared for all the SVIs per chip, hence, saving a lot of hardware resources and enabling ACLs to scale to a larger configurations.

Larger deployments are benefited, where IPv4 Egress Sharing is applied on multiple SVIs with member interfaces on same forwarding ASIC. For example, a trunk port carrying multiple VLANs, and when Egress Sharing is applied on all VLANs it occupies lesser hardware resources irrespective of number of VLANs. By default, IPv4 Egress Sharing is enabled on the switches. However, both IPv4 Egress Sharing and uRPF cannot be enabled at the same time. Disabling IPv4 RACL sharing will allow uRPF configuration and make sure RACL configuration, non-shared mode, is configured at the same time.

To enable unicast Reverse Path Forwarding (uRPF) on the switch, the IPv4 Egress Sharing must me disabled using the `no hardware access-list resource sharing vlan ipv4 out` command.

To enable IPv4 Egress Sharing if previously disabled from the default configuration, use `hardware access-list resource sharing vlan ipv4 out` command. Note, enabling or disabling the IPv4 Egress Sharing requires the restart of software agents on the switches which is a disruptive process and will impact the traffic forwarding.

The following show commands can be used to verify the IPv4 Egress Sharing information on the switch.

- `show ip access-lists`
- `show vlan`
- `show platform arad acl tcam`
- `show ip route`
- `show platform arad ip route`

24.4.3 Configuring IPv4 Egress Sharing

Use `hardware access-list resource sharing vlan ipv4 out` command to enable the IPv4 Egress Sharing on the switch. By default, IPv4 Egress Sharing is enabled on the switch. The `no` form of the command disables the IPv4 Egress Sharing on the switch and user is allowed to configure the uRPF on the switch.
24.4.4 Displaying IPv4 Egress Sharing Information

Examples

- The `show ip access-lists` command displays the list of all the configured IPv4 ACLs.

```plaintext
switch#show ip access-lists summary
IPv4 ACL default-control-plane-acl [readonly]
Total rules configured: 17
Configured on Ingress: control-plane(default VRF)
Active on Ingress: control-plane(default VRF)

IPV4 ACL ipAclLimitTest
Total rules configured: 0
Configured on Egress: Vl2148,2700
Active on Egress: Vl2148,2700
```

- The `show vlan` command displays the list of all the member interfaces under each SVI.

```plaintext
switch#show vlan
VLAN Name                             Status    Ports
----- -------------------------------- --------- -------------------------------
1     default                           active
2148  VLAN2148                          active    Cpu, Et1, Et26
2700  VLAN2700                          active    Cpu, Et18
```

- The `show platform arad acl tcam` command displays the number of TCAM entries (hardware resources) occupied by the ACL on each forwarding ASIC and the percentage of TCAM utilization per forwarding ASIC.

```plaintext
switch#show platform arad acl tcam detail
ip access-list ipAclLimitTest (Shared RACL, 0 rules, 1 entries, direction out, state success, Acl Label 2)
Fap: Arad0, Shared: true, Interfaces: Vl2148, Vl2700
Bank Offset Entries
0 0 1
Fap: Arad1, Shared: true, Interfaces: Vl2148
Bank Offset Entries
0 0 1

Switch#show platform arad acl tcam summary
The total number of TCAM lines per bank is 1024.
==============================================================================================
Arad0:
==============================================================================================
Bank  Used Used % Used By
0 1 0 IP Egress PACLs/RACLs
Total Number of TCAM lines used is: 1
==============================================================================================
Arad1:
==============================================================================================
Bank  Used Used % Used By
0 1 0 IP Egress PACLs/RACLs
Total Number of TCAM lines used is: 1
```
The `show ip route` command displays the unicast ip routes installed in the system.

```
switch#show ip route
VRF name: default
Codes: C - connected, S - static, K - kernel,
       O - OSPF, IA - OSPF inter area, E1 - OSPF external type 1,
       E2 - OSPF external type 2, N1 - OSPF NSSA external type 1,
       N2 - OSPF NSSA external type2, B I - iBGP, B E - eBGP,
       R - RIP, I - ISIS, A B - BGP Aggregate, A O - OSPF Summary,
       NG - Nexthop Group Static Route

Gateway of last resort is not set
C 10.1.0.0/16 is directly connected, Vlan2659
C 10.2.0.0/16 is directly connected, Vlan2148
C 10.3.0.0/16 is directly connected, Vlan2700
S 172.17.0.0/16 [1/0] via 172.24.0.1, Management1
S 172.18.0.0/16 [1/0] via 172.24.0.1, Management1
S 172.19.0.0/16 [1/0] via 172.24.0.1, Management1
S 172.20.0.0/16 [1/0] via 172.24.0.1, Management1
S 172.22.0.0/16 [1/0] via 172.24.0.1, Management1
C 172.24.0.0/18 is directly connected, Management1
```
The `show platform arad ip route` command displays the platform unicast forwarding routes.

```
switch# show platform arad ip route
Tunnel Type: M(mpls), G(gre)

<table>
<thead>
<tr>
<th>VRID</th>
<th>Destination</th>
<th>Cmd</th>
<th>Acl</th>
<th>MAC / CPU Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.0.0.0/8</td>
<td>TRAP</td>
<td>ArpTrap</td>
<td>1031</td>
</tr>
<tr>
<td>0</td>
<td>10.1.0.16</td>
<td>TRAP</td>
<td>ArpTrap</td>
<td>1030</td>
</tr>
<tr>
<td>0</td>
<td>10.2.0.16</td>
<td>TRAP</td>
<td>ArpTrap</td>
<td>1026</td>
</tr>
<tr>
<td>0</td>
<td>10.3.0.16</td>
<td>TRAP</td>
<td>ArpTrap</td>
<td>1034</td>
</tr>
<tr>
<td>0</td>
<td>127.0.0.8</td>
<td>TRAP</td>
<td>ArpTrap</td>
<td>1031</td>
</tr>
<tr>
<td>0</td>
<td>172.17.0.16</td>
<td>TRAP</td>
<td>ArpTrap</td>
<td>1025</td>
</tr>
<tr>
<td>0</td>
<td>172.18.0.16</td>
<td>TRAP</td>
<td>ArpTrap</td>
<td>1025</td>
</tr>
<tr>
<td>0</td>
<td>172.19.0.16</td>
<td>TRAP</td>
<td>ArpTrap</td>
<td>1025</td>
</tr>
<tr>
<td>0</td>
<td>172.20.0.16</td>
<td>TRAP</td>
<td>ArpTrap</td>
<td>1025</td>
</tr>
<tr>
<td>0</td>
<td>172.22.0.16</td>
<td>TRAP</td>
<td>ArpTrap</td>
<td>1025</td>
</tr>
<tr>
<td>0</td>
<td>172.24.0.18</td>
<td>TRAP</td>
<td>ArpTrap</td>
<td>1032</td>
</tr>
</tbody>
</table>
```

This table shows the platform unicast forwarding routes with their respective VRID, destination, command, and MAC/CPU code.
24.5 Route Maps

A route map is an ordered set of rules that control the redistribution of IP routes into a protocol domain on the basis of such criteria as route metrics, access control lists, next hop addresses, and route tags. Route maps can also alter parameters of routes as they are redistributed.

These sections describe the route map implementation:

- Section 24.5.1 describes route maps.
- Section 24.5.2 describes the route map configuration process.
- Section 24.5.3 describes the usage of route maps.

24.5.1 Route Map Description

Route maps are composed of route map statements, each of which consists of a list of match and set commands.

Route Map Statements

Route map statements are categorized by the resolution of routes that the statement filters.

- Permit statements facilitate the redistribution of matched routes.
- Deny statements prevent the redistribution of matched routes.

Route map statement elements include name, sequence number, filter type, match commands, set commands, and continue commands.

- **name** identifies the route map to which the statement belongs.
- **sequence number** designates the statement’s placement within the route map.
- **filter type** specifies the route resolution. Valid types are **permit** and **deny**.
- **match commands** specify criteria that select routes that the statement is evaluating for redistribution.
- **set commands** modify route parameters for redistributed routes.
- **continue commands** prolong the route map evaluation of routes that match a statement.

Statements filter routes for redistribution. Routes that statements pass are redistributed (permit statements) or rejected (deny statements). Routes that statements fail are filtered by the next statement in the route map.

- When a statement does not contain a **match** command, the statement passes all routes.
- When a statement contains a single **match** command that lists a single object, the statement passes routes whose parameters match the object.
- When a statement contains a single **match** command that lists multiple objects, the statements pass routes whose parameters match at least one object.
- When a statement contains multiple **match** commands, the statement passes routes whose parameters match all match commands.

**Set** commands modify parameters for redistributed routes. Set commands are valid in permit statements.
Example

- The following route map statement is named MAP_1 with sequence number 10. The statement matches all routes from BGP Autonomous System 10 and redistributes them with a local preference set to 100. Routes that do not match the statement are evaluated against the next statement in the route map.

  switch#route-map MAP_1 permit 10
  match as 10
  set local-preference 100

Route Maps with Multiple Statements

A route map consists of statements with the same name and different sequence numbers. Statements filter routes in ascending order of their sequence numbers. When a statements passes a route, the redistribution action is performed as specified by the filter type and all subsequent statements are ignored. When the statement fails the route, the statement with the smallest sequence number that is larger than the current one filters the route.

All route maps have an implied final statement that contains a single deny statement with no match command. This denies redistribution to routes that are not passed by any statement.

Example

- The following route map is named MAP_1 with two permit statements. Routes that do not match either statement are denied redistribution into the target protocol domain.

  switch#route-map MAP_1 permit 10
  match as 10
  set local-preference 100

  switch#route-map MAP_1 permit 20
  match metric-type type-1
  match as 100

Section 24.5.2 describes route map configuration procedures.

Route Maps with Multiple Statements and Continue Commands

Route map statements that contain a `continue (route map)` command support additional route map evaluation of routes whose parameters meet the statement’s match commands. Routes that match a statement containing a `continue` command are evaluated against the statement specified by the `continue` command.

When a route matches multiple route map statements, the filter action (deny or permit) is determined by the last statement that the route matches. The `set` commands in all statements matching the route are applied to the route after the route map evaluation is complete. Multiple set commands are applied in the same order by which the route was evaluated against the statements containing them.
Example

- The following route map is named MAP_1 with a permit statement and a deny statement. The permit statement contains a continue command. Routes that match statement 10 are evaluated against statement 20.

```plaintext
route-map MAP_2 permit 10
  match as 10
  continue 20
  set local-preference 100
!
route-map MAP_2 deny 20
  match metric-type type-1
  match as 100
```

The route is redistributed if it passes statement 10 and is rejected by statement 20. The route is denied redistribution in all other instances. The `continue` command guarantees the evaluation of all routes against both statements.

24.5.2 Route Map Configuration

Route maps are created and modified in route map configuration mode. These sections describe the configuration mode and its commands.

- Section 24.5.2.1: Route Map Creation and Editing
- Section 24.5.2.2: Modifying Route Map Components

24.5.2.1 Route Map Creation and Editing

Creating a Route Map Statement

To create a route map, enter `route-map` followed by the map name and filter type (deny or permit). The default sequence number is assigned to the statement if the command does not include a number.

Example

- This command places the switch in route map configuration mode to create a route map statement named `map1` with a sequence number of 50.

```plaintext
switch(config)#route-map map1 permit 50
switch(config-route-map-map1)#
```

Editing a Route Map Statement

To edit an existing route map statement, enter `route-map` with the map’s name and statement’s number. The switch enters route map configuration mode for the statement. Subsequent `match (route-map)` and `set (route-map)` commands add the corresponding commands to the statement.

The `show` command displays contents of the existing route map.
Example

- This command places the switch in route map configuration mode to edit an existing route map statement. The `show` command displays contents of all statements in the route map.

```
switch(config)#route-map MAP2
switch(config-route-map-MAP2)#show
  Match clauses:
    match as 10
    match tag 333
  Set clauses:
    set local-preference 100
switch(config-route-map-MAP2)#
```

Saving Route Map Modifications

Route map configuration mode is a group-change mode. Changes are saved by exiting the mode, either with an explicit `exit` command or by switching directly to another configuration mode. This includes switching to the configuration mode for a different route map.

Example

- The first command creates the `map1` statement with sequence number of 10. The second command is not yet saved to the route map, as displayed by the `show` command.

```
switch(config)#route-map map1 permit
switch(config-route-map-map1)#match as 100
switch(config-route-map-map1)#show

switch(config-route-map-map1)#
```

The `exit` command saves the `match` command.

```
switch(config-route-map-map1)#exit
switch(config)#show route-map map1
route-map map1 permit 10
  Match clauses:
    match as 100
  Set clauses:
switch(config)#
```

Discarding Route Map Modifications

The `abort` command discards all pending changes and exits route map configuration mode.

Example

- The `abort` command discards the pending `match` command and restores the original route map.

```
switch(config)#route-map map1 permit
switch(config-route-map-map1)#match as 100
switch(config-route-map-map1)#abort
switch(config)#show route-map map1
switch(config)#
```

24.5.2.2 Modifying Route Map Components

These commands add rules to the configuration mode route map:

- `match (route-map)` adds a match rule to a route map.
- `set (route-map)` adds a set rule to a route map.
Inserting a Statement

To insert a new statement into an existing route map, create a new statement with a sequence number that differs from any existing statement in the map.

Example

- This command adds statement 50 to the Map1 route map, then displays the new route map.

  switch(config)#route-map Map1 permit 50
  switch(config-route-map-Map1)#match as 150
  switch(config-route-map-Map1)#exit
  switch(config)#show route-map Map1
  route-map Map1 deny 10
    Match clauses:
      match 10
      match tag 333
    Set clauses:
      set local-preference 100
  route-map Map1 permit 50
    Match clauses:
      match as 150
    Set clauses:

Deleting Route Map Components

To remove a component from a route map, perform one of the following:

- To remove a command from a statement, enter no, followed by the command to be removed.
- To remove a statement, enter no, followed by the route map with the filter type and the sequence number of the statement to be removed.
- To remove a route map, enter no followed by the route map without a sequence number.

24.5.3 Using Route Maps

Protocol redistribution commands include a route map parameter that determines the routes to be redistributed into the specified protocol domain.

Example

- This command uses Map1 route map to select OSPFv2 routes for redistribution into BGP AS1.

  switch(config)#router bgp 1
  switch(config-router-bgp)#redistribute ospf route-map Map1
  switch(config-router-bgp)#exit
  switch(config)#

24.6 Prefix Lists

A prefix list is an ordered set of rules that defines route redistribution access for a specified IP address space. A prefix list rules consists of a filter action (deny or permit), an address space identifier (IPv4 subnet address or IPv6 prefix), and a sequence number.

Prefix lists are referenced by route map match commands when filtering routes for redistribution.

- Section 24.6.1 describes the prefix list configuration process.
- Section 24.6.2 describes the use of prefix lists.
24.6.1 Prefix List Configuration

A prefix list is an ordered set of rules that defines route redistribution access for a specified IP address space. A prefix list rule consists of a filter action (deny or permit), a network address (IPv4 subnet or IPv6 prefix), and a sequence number. A rule may also include an alternate mask size.

The switch supports IPv4 and IPv6 prefix lists. The switch is placed in a Prefix-list configuration mode to create and edit IPv4 or IPv6 prefix lists.

24.6.1.1 IPv4 Prefix Lists

IPv4 prefix lists are created or modified by adding an IPv4 prefix list rule in the Prefix-list configuration mode. Each rule includes the name of a prefix list, in addition to the sequence number, network address, and filter action. A list consists of all rules that have the same prefix list name.

The `ip prefix-list` command creates a prefix list or adds a rule to an existing list. Route map match commands use prefix lists to filter routes for redistribution into OSPF, RIP, or BGP domains.

Creating an IPv4 Prefix List

To create an IPv4 prefix list, enter the `ip prefix-list` command, followed by the name of the list. The switch enters IPv4 prefix-list configuration mode for the list. If the command is followed by the name of an existing ACL, subsequent commands edit that list.

Example

- This command places the switch in IPv4 prefix list configuration mode to create an IPv4 prefix list named `route-one`.
  ```
  switch(config)#ip prefix-list route-one
  switch(config-ip-pfx)#
  ```

- These commands create four different rules for the prefix-list named `route-one`.
  ```
  switch(config) #ip prefix-list route-one
  switch(config-ip-pfx) #seq 10 deny 10.1.1.0/24
  switch(config-ip-pfx) #seq 20 deny 10.1.0.0/16
  switch(config-ip-pfx) #seq 30 permit 12.15.4.9/32
  switch(config-ip-pfx) #seq 40 deny 1.1.1.0/24
  ```

To view the list, save the rules by exiting the Prefix-list command mode, then re-enter the configuration mode and type `show active`.

  ```
  switch(config-ip-pfx) #exit
  switch(config) #ip prefix-list route-one
  switch(config-ip-pfx) #show active
  ip prefix-list route-one
    seq 10 deny 10.1.1.0/24
    seq 20 deny 10.1.0.0/16
    seq 30 permit 12.15.4.9/32
    seq 40 deny 1.1.1.0/24
  ```

IPv4 prefix lists are referenced in route map `match (route-map)` commands.

24.6.1.2 IPv6 Prefix Lists

Creating an IPv6 Prefix List

The switch provides IPv6 prefix-list configuration mode for creating and modifying IPv6 prefix lists. A list can be edited only in the mode where it was created.
To create an IP ACL, enter the `ipv6 prefix-list` command, followed by the name of the list. The switch enters IPv6 prefix-list configuration mode for the list. If the command is followed by the name of an existing ACL, subsequent commands edit that list.

**Example**

- This command places the switch in IPv6 prefix list configuration mode to create an IPv6 prefix list named `map1`.

  ```
  switch(config)#ipv6 prefix-list map1
  switch(config-ipv6-pfx)#
  ```

**Adding a Rule**

To append a rule to the end of a list, enter the rule without a sequence number while in Prefix-List configuration mode for the list. The new rule’s sequence number is derived by adding 10 to the last rule’s sequence number.

**Example**

- These commands enter the first two rules into a new prefix list.

  ```
  switch(config-ipv6-pfx)#permit 3:4e96:8ca1:33cf::/64
  switch(config-ipv6-pfx)#permit 3:11b1:8fe4:1aac::/64
  ```

  To view the list, save the rules by exiting the prefix-list command mode, then re-enter the configuration mode and type `show active`.

  ```
  switch(config-ipv6-pfx)#exit
  switch(config)#ipv6 prefix-list map1
  switch(config-ipv6-pfx)#show active
  ipv6 prefix-list map1
  seq 10 permit 3:4e96:8ca1:33cf::/64
  seq 20 permit 3:11b1:8fe4:1aac::/64
  switch(config-ipv6-pfx)#
  ```

  This command appends a rule to the end of the prefix list. The new rule’s sequence number is 30.

  ```
  switch(config-ipv6-pfx)#permit 3:1bca:1141:ab34::/64
  switch(config-ipv6-pfx)#exit
  switch(config)#ipv6 prefix-list map1
  switch(config-ipv6-pfx)#show active
  ipv6 prefix-list map1
  seq 10 permit 3:4e96:8ca1:33cf::/64
  seq 20 permit 3:11b1:8fe4:1aac::/64
  seq 30 permit 3:1bca:1141:ab34::/64
  switch(config-ipv6-pfx)#
  ```

**Inserting a Rule**

To insert a rule into a prefix list, use the `seq` (IPv6 Prefix Lists) command to enter a rule with a sequence number that is between numbers of two existing rules.
Example

- This command inserts a rule between the first two rules by assigning it the sequence number 15.

    switch(config-ipv6-pfx)#seq 15 deny 3:4400::/64
    switch(config-ipv6-pfx)#exit
    switch(config)#show ipv6 prefix-list map1
    ipv6 prefix-list map1
    seq 10 permit 3:4e96:8ca1:33cf::/64
    seq 15 deny 3:4400::/64
    seq 20 permit 3:11b1:8fe4:1aac::/64
    seq 30 permit 3:1bca:3ff2:634a::/64
    switch(config)#

Deleting a Rule

To remove a rule from the configuration mode prefix list, enter `no seq` (see `seq (IPv6 Prefix Lists)`), followed by the sequence number of the rule to be removed.

Example

- These commands remove rule 20 from the prefix list, then displays the resultant prefix list.

    switch(config-ipv6-pfx)#no seq 20
    switch(config-ipv6-pfx)#exit
    switch(config)#show ipv6 prefix-list map1
    ipv6 prefix-list map1
    seq 10 permit 3:4e96:8ca1:33cf::/64
    seq 15 deny 3:4400::/64
    seq 30 permit 3:1bca:3ff2:634a::/64
    switch(config)#

24.6.2 Using Prefix Lists

Route map match commands include an option that matches a specified prefix list.

Example

- The MAP_1 route map uses a match command that references the PL_1 prefix list.

    switch(config)#route-map MAP_1 permit
    switch(config-route-map-MAP_1)#match ip address prefix-list PL_1
    switch(config-route-map-MAP_1)#set community 500
    switch(config-route-map-MAP_1)#exit
24.7 **ACL, Route Map, and Prefix List Commands**

This section describes CLI commands that this chapter references.

**ACL Creation and Access Commands**
- `ip access-list`
- `ip access-list standard`
- `ipv6 access-list`
- `ipv6 access-list standard`
- `mac access-list`
- `system profile`
- `hardware access-list resource sharing vlan in`
- `hardware access-list resource sharing vlan ipv4 out`

**ACL Implementation Commands**
- `ip access-group`
- `ipv6 access-group`
- `mac access-group`

**Service ACL Implementation Commands**
- `ip access-group (Service ACLs)`
- `ipv6 access-group (Service ACLs)`

**ACL Edit Commands**
- `counters per-entry (ACL configuration modes)`
- `hardware access-list update default-result permit`
- `no <sequence number> (ACLs)`
- `resequence (ACLs)`
- `show (ACL configuration modes)`

**ACL Rule Commands**
- `deny (IPv4 ACL)`
- `deny (IPv6 ACL)`
- `deny (MAC ACL)`
- `deny (Standard IPv4 ACL)`
- `deny (Standard IPv6 ACL)`
- `permit (IPv4 ACL)`
- `permit (IPv6 ACL)`
- `permit (MAC ACL)`
- `permit (Standard IPv4 ACL)`
- `permit (Standard IPv6 ACL)`
- `remark`

**ACL List Counter Commands**
- `clear ip access-lists counters`
- `clear ipv6 access-lists counters`
- `hardware counter feature acl out`

**ACL Display Commands**
- `show ip access-lists`
- `show ipv6 access-lists`
- `show mac access-lists`
Prefix List Creation and Access Commands
- ip prefix-list
- ipv6 prefix-list

Prefix List Edit Commands
- deny (IPv6 Prefix List)
- permit (IPv6 Prefix List)
- seq (IPv6 Prefix Lists)

Prefix List Display Commands
- show ip prefix-list
- show ipv6 prefix-list
- show platform trident tcam
- show platform fap acl
- show platform fap acl tcam
- show platform arad acl tcam
- show platform arad acl tcam summary
- show platform arad mapping
- show hardware tcam profile
- show platform fap acl tcam hw

Route Map Creation and Access Command
- route-map

Route Map Edit Commands
- continue (route map)
- description (route map)
- match (route-map)
- set (route-map)
- set as-path prepend
- set as-path match
- set community (route-map)
- set extcommunity (route-map)

Route Map Display Commands
- show route-map
clear ip access-lists counters

The **clear ip access-lists counters** command sets ACL counters to zero for the specified IPv4 access control list (ACL). The **session** parameter limits ACL counter clearing to the current CLI session.

**Command Mode**

Privileged EXEC

**Command Syntax**

```plaintext
clear ip access-lists counters [ACL_NAME] [SCOPE]
```

**Parameters**

- **ACL_NAME** Name of ACL. Options include:
  - <no parameter> all ACLs.
  - access_list name of ACL.
- **SCOPE** Session affected by command. Options include:
  - <no parameter> command affects counters on all CLI sessions.
  - session affects only current CLI session.

**Example**

- This command resets all IPv4 ACL counters.
  ```plaintext
  switch(config)#clear ip access-lists counters
  switch(config)#
  ```
clear ipv6 access-lists counters

The `clear ipv6 access-lists counters` command sets ACL counters to zero for the specified IPv6 access control list (ACL). The `session` parameter limits ACL counter clearing to the current CLI session.

**Command Mode**

Privileged EXEC

**Command Syntax**

```
clear ipv6 access-lists counters [ACL_NAME] [SCOPE]
```

**Parameters**

- **ACL_NAME** name of ACL. Options include:
  - `<no parameter>` all IPv6 ACLs.
  - `access_list` name of IPv6 ACL.

- **SCOPE** Session affected by command. Options include:
  - `<no parameter>` command affects counters on all CLI sessions.
  - `session` affects only current CLI session.

**Example**

- This command resets all IPv6 ACL counters.

```
switch(config)# clear ipv6 access-lists counters
switch(config)#
```
**continue (route map)**

The `continue` command creates a route map statement entry that enables additional route map evaluation of routes whose parameters meet the statement’s matching criteria.

A statement typically contains a `match (route-map)` and a `set (route-map)` command. The evaluation of routes whose settings are the same as `match` command parameters normally ends and the statement’s `set` commands are applied to the route. Routes that match a statement containing a `continue` command are evaluated against the statement specified by the `continue` command.

When a route matches multiple route map commands, the filter action (deny or permit) is determined by the last statement that the route matches. The `set` commands in all statements matching the route are applied to the route after the route map evaluation is complete. Multiple `set` commands are applied in the same order by which the route was evaluated against the statement containing them.

The `no continue` and `default continue` commands remove the corresponding `continue` command from the configuration mode route map statement by deleting the corresponding command from `running-config`.

**Command Mode**

Route-Map Configuration

**Command Syntax**

```
continue NEXT_SEQ
no continue NEXT_SEQ
default continue NEXT_SEQ
```

**Parameters**

- `NEXT_SEQ` specifies next statement for evaluating matching routes. Options include:
  - `<no parameter>` Next statement in the route map, as determined by sequence number.
  - `seq_number` Specifies the number of the next statement. Values range from 1 to 16777215.

**Restrictions**

A `continue` command cannot specify a sequence number smaller than the sequence number of its route map statement.

**Related Commands**

- `route-map` enters route map configuration mode.

**Example**

- This command creates route map map1, statement 40 with a `match` command, a `set` command, and a `continue` command. Routes that match the statement are subsequently evaluated against statement 100. The `set local-preference` command is applied to matching routes regardless of subsequent matching operations.
  
  ```
  switch(config)#route-map map1 deny 40
  switch(config-route-map-map1)#match as 15
  switch(config-route-map-map1)#continue 100
  switch(config-route-map-map1)#set local-preference 50
  switch(config-route-map-map1)#
  ```
counters per-entry (ACL configuration modes)

The **counters per-entry** command places the ACL in counting mode. An ACL in counting mode displays the number of instances each rule in the list matches an inbound packet and the elapsed time since the last match. The show access list commands display the statistics next to each rule in the ACL.

On the FM6000 platform, this command has no effect when used in an ACL that is part of a PBR class map.

The **no counters per-entry** and **default counters per-entry** command places the ACL in non-counting mode.

**Command Mode**

- ACL Configuration
- IPv6-ACL Configuration
- Std-ACL Configuration
- Std-IPv6-ACL Configuration
- MAC-ACL Configuration

**Command Syntax**

- `counters per-entry`
- `no counters per-entry`
- `default counters per-entry`

**Examples**

- This command places the test1 ACL in counting mode.

  ```
  switch(config)#ip access-list test1
  switch(config-acl-test1)#counters per-entry
  switch(config-acl-test1)#
  ```

- This command displays the ACL, with counter information, for an ACL in counting mode.

  ```
  switch#show ip access-lists
  IP Access List default-control-plane-acl [readonly]
  counters per-entry
  10 permit icmp any any
  20 permit ip any any tracked [match 12041, 0:00:00 ago]
  30 permit ospf any any
  40 permit tcp any any eq ssh telnet www snmp bgp https [match 11, 1:41:07 ago]
  50 permit udp any any bootps bootpc snmp rip [match 78, 0:00:27 ago]
  60 permit tcp any any range 5900 5910
  70 permit udp any any range 50000 50100
  80 permit vrrp any any
  90 permit ahp any any
  100 permit pim any any
  110 permit igmp any any [match 14, 0:23:27 ago]
  120 permit tcp any any range 5900 5910
  130 permit tcp any any range 50000 50100
  140 permit udp any any range 51000 51100
  ```
deny (IPv4 ACL)

The `deny` command adds a deny rule to the configuration mode IPv4 access control list (ACL). Packets filtered by a `deny` rule are dropped by interfaces to which the ACL is applied. Sequence numbers determine rule placement in the ACL. Sequence numbers for commands without numbers are derived by adding 10 to the number of the ACL’s last rule.

The `no deny` and `default deny` commands remove the specified rule from the configuration mode ACL. The `no <sequence number> (ACLs)` command also removes the specified rule from the ACL.

Command Mode

ACL Configuration

Command Syntax

```plaintext
[SEQ_NUM] deny PROTOCOL SOURCE_ADDR [SOURCE_PORT] DEST_ADDR [DEST_PORT] [FLAGS] [MESSAGE] [fragments] [tracked] [DSCP_FILTER] [TTL_FILTER] [log]

no deny PROTOCOL SOURCE_ADDR [SOURCE_PORT] DEST_ADDR [DEST_PORT] [FLAGS] [MESSAGE] [fragments] [tracked] [DSCP_FILTER] [TTL_FILTER] [log]

default deny PROTOCOL SOURCE_ADDR [SOURCE_PORT] DEST_ADDR [DEST_PORT] [FLAGS] [MESSAGE] [fragments] [tracked] [DSCP_FILTER] [TTL_FILTER] [log]
```

Commands use a subset of the listed fields. Available parameters depend on specified protocol. Use CLI syntax assistance to view options for specific protocols when creating a deny rule.

Parameters

- **SEQ_NUM**  
  Sequence number assigned to the rule. Options include:
  - `<no parameter>`  
    Number is derived by adding 10 to the number of the ACL’s last rule.
  - `<1 – 4294967295>`  
    Number assigned to entry.

- **PROTOCOL**  
  protocol field filter. Values include:
  - `ahp`  
    Authentication Header Protocol (51).
  - `icmp`  
    Internet Control Message Protocol (1).
  - `igmp`  
    Internet Group Management Protocol (2).
  - `ip`  
    Internet Protocol v4 (4).
  - `ospf`  
    Open Shortest Path First (89).
  - `pim`  
    Protocol Independent Multicast (103).
  - `tcp`  
    Transmission Control Protocol (6).
  - `udp`  
    user datagram protocol (17).
  - `vrrp`  
    Virtual Router Redundancy Protocol (112).
  - `protocol_num`  
    integer corresponding to an IP protocol. Values range from 0 to 255.

- **SOURCE_ADDR** and **DEST_ADDR**  
  source and destination address filters. Options include:
  - `network_addr`  
    subnet address (CIDR or address-mask).
  - `any`  
    Packets from all addresses are filtered.
  - `host ip_addr`  
    IP address (dotted decimal notation).
    Subnet addresses support discontiguous masks.

- **SOURCE_PORT** and **DEST_PORT**  
  source and destination port filters. Options include:
• **any**  all ports
• **eq**  `port-1`  `port-2`  ...  `port-n`  A list of ports. Maximum list size is 10 ports.
• **neq**  `port-1`  `port-2`  ...  `port-n`  The set of all ports not listed. Maximum list size is 10 ports.
• **gt**  `port`  The set of ports with larger numbers than the listed port.
• **lt**  `port`  The set of ports with smaller numbers than the listed port.
• **range**  `port_1`  `port_2`  The set of ports whose numbers are between the range.
• **fragments**  filters packets with FO bit set (indicates a non-initial fragment packet).
• **FLAGS**  flag bit filters (TCP packets). Use CLI syntax assistance (?) to display options.
• **MESSAGE**  message type filters (ICMP packets). Use CLI syntax assistance (?) to display options.
• **tracked**  rule filters packets in existing ICMP, UDP, or TCP connections.
  • Valid in ACLs applied to the control plane.
  • Validity in ACLs applied to data plane varies by switch platform.
• **DSCP_FILTER**  rule filters packet by its DSCP value. Values include:
  • <no parameter>  Rule does not use DSCP to filter packets.
  • **dscp**  `dscp_value`  Packets match if DSCP field in packet is equal to `dscp_value`.
• **TTL_FILTER**  rule filters packet by its TTL (time-to-live) value. Values include:
  • **ttl eq**  `ttl_value`  Packets match if `ttl` in packet is equal to `ttl_value`.
  • **ttl gt**  `ttl_value`  Packets match if `ttl` in packet is greater than `ttl_value`.
  • **ttl lt**  `ttl_value`  Packets match if `ttl` in packet is less than `ttl_value`.
  • **ttl neq**  `ttl_value`  Packets match if `ttl` in packet is not equal to `ttl_value`.
  • Valid in ACLs applied to the control plane.
  • Validity in ACLs applied to data plane varies by switch platform.
• **log**  triggers an informational log message to the console about the matching packet.
  • Valid in ACLs applied to the control plane.
  • Validity in ACLs applied to data plane varies by switch platform.

**Examples**

• This command appends a **deny** statement at the end of the ACL. The **deny** statement drops OSPF packets from 10.10.1.1/24 to any host.
  ```
  switch(config)#ip access-list text1
  switch(config-acl-text1)#deny ospf 10.1.1.0/24 any
  switch(config-acl-text1)#
  ```
• This command inserts a **deny** statement with the sequence number 65. The **deny** statement drops all PIM packets.
  ```
  switch(config-acl-text1)#65 deny pim any any
  switch(config-acl-text1)#
  ```
deny (IPv6 ACL)

The `deny` command adds a deny rule to the configuration mode IPv6 access control list (ACL). Packets filtered by a `deny` rule are dropped by interfaces to which the ACL is applied. Sequence numbers determine rule placement in the ACL. Sequence numbers for commands without numbers are derived by adding 10 to the number of the ACL’s last rule.

The `no deny` and `default deny` commands remove the specified rule from the configuration mode ACL. The `no <sequence number>` (ACLs) command also removes the specified rule from the ACL.

Command Mode
IPv6-ACL Configuration

Command Syntax

```
[SEQ_NUM] deny PROT SRC_ADDR [SRC_PT] DEST_ADDR [DEST_PT] [FLAG] [MSG] [HOP]
[tracked] [DSCP_FILTER] [log]
```

```
no deny PROT SRC_ADDR [SRC_PT] DEST_ADDR [DEST_PT] [FLAG] [MSG] [HOP] [tracked]
[DSCP_FILTER] [log]
```

```
default deny PROT SRC_ADDR [SRC_PT] DEST_ADDR [DEST_PT] [FLAG] [MSG] [HOP] [tracked]
[DSCP_FILTER] [log]
```

Commands use a subset of the listed fields. Available parameters depend on specified protocol. Use CLI syntax assistance to view options for specific protocols when creating a deny rule.

Parameters

- **SEQ_NUM**  Sequence number assigned to the rule. Options include:
  - `<no parameter>`  Number is derived by adding 10 to the number of the ACL’s last rule.
  - `<1 – 4294967295>`  Number assigned to entry.

- **PROT**  protocol field filter. Values include:
  - `icmpv6`  Internet Control Message Protocol for version 6 (58).
  - `ipv6`  Internet Protocol – IPv6 (41).
  - `ospf`  Open Shortest Path First (89).
  - `tcp`  Transmission Control Protocol (6).
  - `udp`  User Datagram Protocol (17).

- **SRC_ADDR** and **DEST_ADDR**  source and destination address filters. Options include:
  - `ipv6_prefix`  IPv6 address with prefix length (CIDR notation).
  - `any`  Packets from all addresses are filtered.
  - `host ipv6_addr`  IPv6 host address.

- **SRC_PT** and **DEST_PT**  source and destination port filters. Options include:
  - `any`  all ports.
  - `eq port-1 port-2 ... port-n`  A list of ports. Maximum list size is 10 ports.
  - `neq port-1 port-2 ... port-n`  The set of all ports not listed. Maximum list size is 10 ports.
  - `gt port`  The set of ports with larger numbers than the listed port.
  - `lt port`  The set of ports with smaller numbers than the listed port.
  - `range port_1 port_2`  The set of ports whose numbers are between the range.
• **HOP** filters by packet’s hop-limit value. Options include:
  • <no parameter> Rule does not use hop limit to filter packets.
  • **hop-limit eq** hop_value Packets match if **hop-limit** value in packet equals **hop_value**.
  • **hop-limit gt** hop_value Packets match if **hop-limit** in packet is greater than **hop_value**.
  • **hop-limit lt** hop_value Packets match if **hop-limit** in packet is less than **hop_value**.
  • **hop-limit neq** hop_value Packets match if **hop-limit** in packet is not equal to **hop_value**.

• **FLAG** flag bit filters (TCP packets). Use CLI syntax assistance (?) to display options.

• **MSG** message type filters (ICMPv6 packets). Use CLI syntax assistance (?) to display options.

• **tracked** rule filters packets in existing ICMP, UDP, or TCP connections.
  • Valid in ACLs applied to the control plane.
  • Validity in ACLs applied to data plane varies by switch platform.

• **DSCP_FILTER** rule filters packet by its DSCP value. Values include:
  • <no parameter> Rule does not use DSCP to filter packets.
  • **dscp** dscp_value Packets match if DSCP field in packet is equal to **dscp_value**.
  • **log** triggers an informational log message to the console about the matching packet.
  • Valid in ACLs applied to the control plane.
  • Validity in ACLs applied to data plane varies by switch platform.

**Example**

• This command appends a **deny** statement at the end of the ACL. The **deny** statement drops IPv6 packets from 3710:249a:c643:ef11::/64 to any host.

  ```
  switch(config)#ipv6 access-list text1
  switch(config-acl-text1)#deny ipv6 3710:249a:c643:ef11::/64 any
  switch(config-acl-text1)#
  ```
deny (IPv6 Prefix List)

The **deny** command adds a rule to the configuration mode IPv6 prefix list. Route map match commands use prefix lists to filter routes for redistribution into OSPF, RIP, or BGP domains. Routes are denied access when they match the prefix that a **deny** statement specifies.

The **no deny** and **default deny** commands remove the specified rule from the configuration mode prefix list. The **no seq (IPv6 Prefix Lists)** command also removes the specified rule from the prefix list.

**Command Mode**
IPv6-pfx Configuration

**Command Syntax**
```
[SEQUENCE] deny ipv6_prefix [MASK]
```

**Parameters**
- **SEQUENCE** Sequence number assigned to the rule. Options include:
  - <no parameter> Number is derived by adding 10 to the number of the list's last rule.
  - `seq seq_num` Number is specified by `seq_num`. Value ranges from 0 to 65535.
  - `ipv6_prefix` IPv6 prefix upon which command filters routes (CIDR notation).
- **MASK** range of the prefix to be matched.
  - <no parameter> exact match with the subnet mask is required.
  - `eq mask_e` prefix length is equal to `mask_e`.
  - `ge mask_g` range is from `mask_g` to 128.
  - `le mask_l` range is from `subnet` mask length to `mask_l`.
  - `ge mask_l` and `le mask_g` range is from `mask_g` to `mask_l`.
  - `mask_e, mask_l` and `mask_g` range from 1 to 128.

  when `le` and `ge` are specified, `subnet` mask > `mask_g` > `mask_l`

**Example**
- This command appends a **deny** statement at the end of the text1 prefix list. The **deny** statement denies redistribution of routes with the specified prefix.

```
switch(config)#ipv6 prefix-list route-five
switch(config-ipv6-pfx)#deny 3100::/64
switch(config-ipv6-pfx)#
```
deny (MAC ACL)

The `deny` command adds a deny rule to the configuration mode MAC access control list (ACL). Packets filtered by a deny rule are dropped by interfaces to which the ACL is applied. Sequence numbers determine rule placement in the ACL. Sequence numbers for commands without numbers are derived by adding 10 to the number of the ACL’s last rule.

The `no deny` and `default deny` commands remove the specified rule from the configuration mode ACL. The `no <sequence number> (ACLs)` command also removes the specified rule from the ACL.

Command Mode
MAC-ACL Configuration

Command Syntax

```
[SEQ_NUM] deny SOURCE_ADDR DEST_ADDR [PROTOCOL] [log]
no deny SOURCE_ADDR DEST_ADDR [PROTOCOL] [log]
default deny SOURCE_ADDR DEST_ADDR [PROTOCOL] [log]
```

Parameters

- **SEQ_NUM**  Sequence number assigned to the rule. Options include:
  - `<no parameter>` Number is derived by adding 10 to the number of the ACL’s last rule.
  - `<1 – 4294967295>` Number assigned to entry.
- **SOURCE_ADDR** and **DEST_ADDR**  source and destination address filters. Options include:
  - `mac_address mac_mask`  MAC address and mask
  - `any`  Packets from all addresses are filtered.
    - `mac_address` specifies a MAC address in 3x4 dotted hexadecimal notation
      (hhhh.hhhh.hhhh)
    - `mac_mask` specifies a MAC address mask in 3x4 dotted hexadecimal notation
      (hhhh.hhhh.hhhh)
  - 0 bits require an exact match to filter
  - 1 bits filter on any value
- **PROTOCOL**  protocol field filter. Values include:
  - `aarp`  Appletalk Address Resolution Protocol (0x80f3)
  - `appletalk`  Appletalk (0x809b)
  - `arp`  Address Resolution Protocol (0x806)
  - `ip`  Internet Protocol Version 4 (0x800)
  - `ipx`  Internet Packet Exchange (0x8137)
  - `lldp`  LLDP (0x88cc)
  - `novell`  Novell (0x8138)
  - `rarp`  Reverse Address Resolution Protocol (0x8035)
  - `protocol_num`  integer corresponding to a MAC protocol. Values range from 0 to 65535
  - `log`  triggers an informational log message to the console about the matching packet.
Examples

- This command appends a permit statement at the end of the ACL. The deny statement drops all aarp packets from 10.1000.0000 through 10.1000.FFFF to any host.

  switch(config)#mac access-list text1
  switch(config-mac-acl-text1)#deny 10.1000.0000 0.0.FFFF any aarp

- This command inserts a permit statement with the sequence number 25. The deny statement drops all packets through the interface.

  switch(config-mac-acl-text1)#25 deny any any
deny (Standard IPv4 ACL)

The `deny` command adds a deny rule to the configuration mode standard IPv4 access control list (ACL). Standard ACL rules filter on the source field.

Packets filtered by a `deny` rule are dropped by interfaces to which the ACL is applied. Sequence numbers determine rule placement in the ACL. Sequence numbers for commands without numbers are derived by adding 10 to the number of the ACL’s last rule.

The `no deny` and `default deny` commands remove the specified rule from the configuration mode ACL. The `no <sequence number>` (ACLs) command also removes the specified rule from the ACL.

Command Mode

Std-ACL Configuration

Command Syntax

```
[SEQ_NUM] deny SOURCE_ADDR [log]
no deny SOURCE_ADDR [log]
default deny SOURCE_ADDR [log]
```

Parameters

- **SEQ_NUM**  Sequence number assigned to the rule. Options include:
  - `<no parameter>`  Number is derived by adding 10 to the number of the ACL’s last rule.
  - `<1 – 4294967295>`  Number assigned to entry.
- **SOURCE_ADDR**  source address filter. Options include:
  - `network_addr`  subnet address (CIDR or address-mask).
  - `any`  packets from all addresses are filtered.
  - `host ip_addr`  IP address (dotted decimal notation).

Subnet addresses support discontiguous masks.

- **log**  triggers an informational log message to the console about the matching packet.
  - Valid in ACLs applied to the control plane.
  - Validity in ACLs applied to data plane varies by switch platform.

Example

- This command appends a `deny` statement at the end of the ACL. The `deny` statement drops packets from 10.10.1.1/24.

```
switch(config)#ip access-list standard text1
switch(config-std-acl-text1)#deny 10.1.1.1/24
switch(config-std-acl-text1)#
```
**deny (Standard IPv6 ACL)**

The `deny` command adds a deny rule to the configuration mode standard IPv6 access control list (ACL). Standard ACL rules filter on the source field.

Packets filtered by a `deny` rule are dropped by interfaces to which the ACL is applied. Sequence numbers determine rule placement in the ACL. Sequence numbers for commands without numbers are derived by adding 10 to the number of the ACL’s last rule.

The `no deny` and `default deny` commands remove the specified rule from the configuration mode ACL. The `no <sequence number> (ACLs)` command also removes the specified rule from the ACL.

**Command Mode**

Std-IPv6-ACL Configuration

**Command Syntax**

```
[SEQ_NUM] deny SOURCE_ADDR
no deny SOURCE_ADDR
default deny SOURCE_ADDR
```

**Parameters**

- **SEQ_NUM**  Sequence number assigned to the rule. Options include:
  - `<no parameter>` Number is derived by adding 10 to the number of the ACL’s last rule.
  - `<1 – 4294967295>` Number assigned to entry.
- **SOURCE_ADDR**  source address filter. Options include:
  - `ipv6_prefix` IPv6 address with prefix length (CIDR notation).
  - `any` Packets from all addresses are filtered.
  - `host ipv6_addr` IPv6 host address.

**Example**

- This command appends a `deny` statement at the end of the ACL. The `deny` statement drops packets from 2103::/64.

  ```
  switch(config)#ipv6 access-list standard text1
  switch(config-std-acl-ipv6-text1)#deny 2103::/64
  switch(config-std-acl-ipv6-text1)#
  ```
description (route map)

The `description` command adds a text string to the configuration mode route map. The string has no functional impact on the route map.

The `no description` and default `description` commands remove the text string from the configuration mode route map by deleting the corresponding `description` command from `running-config`.

**Command Mode**

Route-Map Configuration

**Command Syntax**

```
description label_text
no description
default description
```

**Parameters**

- `label_text` character string assigned to the route map configuration.

**Related Commands**

- `route-map`

**Examples**

- These commands add description text to the XYZ-1 route map.

```
switch(config)#route-map XYZ-1
switch(config-route-map-XYZ-1)#description This is the first map.
switch(config-route-map-XYZ-1)#exit
switch(config)#show route-map XYZ-1
route-map XYZ-1 permit 10
  Description:
    description This is the first map.
  Match clauses:
  Set clauses:
switch(config)#
```
**hardware access-list update default-result permit**

The `hardware access-list update default-result permit` command configures the switch to permit all traffic on Ethernet and VLAN interfaces with ACLs applied to them while those ACLs are being modified. Traffic is permitted when the ACL is available for modification using one of the `ip access-list` commands, and ends when the ACL configuration mode is exited and rules are populated in hardware. This command is disabled by default.

The `no hardware access-list update default-result permit` and `default hardware access-list update default-result permit` commands restore the switch to its default state (blocking traffic during ACL modifications) by removing the corresponding `hardware access-list update default-result permit` command from the `running-config`.

**Command Mode**
- Global Configuration

**Command Syntax**

```plaintext
hardware access-list update default-result permit
no hardware access-list update default-result permit
default hardware access-list update default-result permit
```

**Restrictions**

This command is available on the Arista 7050X, 7060X, 7150, 7250X, 7280, 7280R, 7300X, 7320X, and 7500 series switches.

This command does not support egress ACLs.

While this command is enabled, static NAT and ACL-based mirroring are affected during ACL updates.

**Example**

- This command configures a 7150 series switch to permit all traffic on Ethernet and VLAN interfaces with ACLs applied to them while those ACLs are being modified.

```
switch(config)#hardware access-list update default-result permit
switch(config)#
```
hardware counter feature acl out

The **hardware counter feature acl out** command enables egress ACL hardware counters for IPv4 or IPv6, which count the number of packets hitting rules associated with egress ACLs applied to various interfaces on a switch.

The **no hardware counter feature acl out** and **default hardware counter feature acl out** commands disable or return the egress ACL hardware counters to the default state.

**Command Mode**

Global Configuration

**Command Syntax**

```
hardware counter feature acl out [OPTIONS]
no hardware counter feature acl out [OPTIONS]
default hardware counter feature acl out [OPTIONS]
```

**Parameters**

- **OPTIONS**
  - **ipv4** address family IPv4.
  - **ipv6** address family IPv6.

**Example**

- This command enables IPv4 egress ACL hardware counters.
  
  ```
  switch(config)#hardware counter feature acl out ipv4
  switch(config)#
  ```

- This command disables IPv4 egress ACL hardware counters.
  
  ```
  switch(config)#no hardware counter feature acl out ipv4
  switch(config)#
  ```
**ip access-group**

The `ip access-group` command applies an IPv4 or standard IPv4 access control list (ACL) to the configuration mode interface or subinterface.

The `no ip access-group` and `default ip access-group` commands remove the corresponding `ip access-group` command from `running-config`.

**Command Mode**
- Interface-Ethernet Configuration
- Interface-Port-Channel Configuration
- Interface-VLAN Configuration

**Command Syntax**
```
ip access-group list_name DIRECTION
no ip access-group list_name DIRECTION
default ip access-group list_name DIRECTION
```

**Parameters**
- `list_name`  name of ACL assigned to interface.
- `DIRECTION` transmission direction of packets, relative to interface. Valid options include:
  - `in`  inbound packets.
  - `out`  outbound packets.

**Restrictions**
Filtering of outbound packets by ACLs is not supported on Petra platform switches.
Filtering of outbound packets by ACLs on FM6000 switches is supported only on physical interfaces (Ethernet and port channels).
ACLs on sub-interfaces are supported on DCS-7280E, DCS-7500E, DCS-7280R, and DCS-7500R.

**Example**
- These commands apply the IPv4 ACL named `test2` to Ethernet interface 3.
  ```
  switch(config)#interface ethernet 3
  switch(config-if-Et3)#ip access-group test2 in
  switch(config-if-Et3)#
  ```
ip access-group (Service ACLs)

The `ip access-group (Service ACLs)` command configures a Service ACL to be applied by a control-plane service. The service is specified by the command mode in which the Service ACL is applied.

The `no ip access-group (Service ACLs)` and `default ip access-group (Service ACLs)` commands remove the corresponding `ip access-group (Service ACLs)` command from `running-config`.

**Command Mode**

- Mgmt-SSH Configuration
- Mgmt-API Configuration
- Router-BGP Configuration
- Router-OSPF Configuration
- Router-IGMP Configuration
- MPLS-LDP Configuration
- Queue-Monitor-Streaming Configuration
- MPLS-Ping Configuration
- Mgmt-Telnet Configuration

**Command Syntax**

```
ip access-group acl_name [vrf vrf_name] [in]
no ip access-group [acl_name] [vrf vrf_name] [in]
default ip access-group acl_name [vrf vrf_name] [in]
```

**Parameters**

Parameters vary by process.

- `acl_name` name of the Service ACL assigned to control-plane service.
- `vrf vrf_name` specifies the VRF in which the Service ACL is to be applied.
- `in` specifies inbound connections or packets only (keyword required for SSH and Telnet services).

**Example**

- These commands apply the Service ACL `bgpacl` to the BGP routing protocol in VRF `purple`.

  ```
  (config)# router bgp 5
  (config-router-bgp)# vrf purple
  (config-router-bgp-vrf-purple)# ip access-group bgpacl
  ```

For additional configuration examples, see [Section 24.3.2: Configuring Service ACLs and Displaying Status and Counters](#).
**ip access-list**

The `ip access-list` command places the switch in ACL configuration mode, which is a group change mode that modifies an IPv4 access control list. The command specifies the name of the IPv4 ACL that subsequent commands modify and creates an ACL if it references a nonexistent list. All changes in a group change mode edit session are pending until the end of the session.

The `exit` command saves pending ACL changes to `running-config`, then returns the switch to global configuration mode. ACL changes are also saved by entering a different configuration mode.

The `abort` command discards pending ACL changes, returning the switch to global configuration mode.

The `no ip access-list` and `default ip access-list` commands delete the specified IPv4 ACL.

**Command Mode**

Global Configuration

**Command Syntax**

```
ip access-list list_name
no ip access-list list_name
default ip access-list list_name
```

**Parameters**

- `list_name` Name of ACL.
  Must begin with an alphabetic character. Cannot contain spaces or quotation marks.

**Commands Available in ACL configuration mode:**

- `deny (IPv4 ACL)`
- `no <sequence number> (ACLs)`
- `permit (IPv4 ACL)`
- `remark`
- `resequence (ACLs)`
- `show (ACL configuration modes)`

**Related Commands**

- `ip access-list standard` enters std-acl configuration mode for editing standard IP ACLs.
- `show ip access-lists` displays IP and standard ACLs.

**Examples**

- This command places the switch in ACL configuration mode to modify the `filter1` IPv4 ACL.
  ```
  switch(config)#ip access-list filter1
  switch(config-acl-filter1)#
  ```
- This command saves changes to `filter1` ACL, then returns the switch to global configuration mode.
  ```
  switch(config-acl-filter1)#exit
  switch(config)#
  ```
- This command discards changes to `filter1`, then returns the switch to global configuration mode.
  ```
  switch(config-acl-filter1)#abort
  switch(config)#
  ```
The `ip access-list standard` command places the switch in std-ACL configuration mode, which is a group change mode that modifies a standard IPv4 access control list. The command specifies the name of the standard IPv4 ACL that subsequent commands modify, and creates an ACL if it references a nonexistent list. All group change mode edit session changes are pending until the session ends.

The `exit` command saves pending ACL changes to `running-config`, then returns the switch to global configuration mode. Pending changes are also saved by entering a different configuration mode.

The `abort` command discards pending ACL changes, returning the switch to global configuration mode.

The `no ip access-list standard` and `default ip access-list standard` commands delete the specified ACL.

**Command Mode**
- Global Configuration

**Command Syntax**

```
ip access-list standard  list_name  
no ip access-list standard  list_name  
default ip access-list standard  list_name  
```

**Parameters**
- `list_name` Name of standard ACL. Must begin with an alphabetic character. Cannot contain spaces or quotation marks.

**Commands Available in std-ACL configuration mode:**
- `deny` (Standard IPv4 ACL)
- `no <sequence number>` (ACLs)
- `permit` (Standard IPv4 ACL)
- `remark`
- `resequence` (ACLs)
- `show` (ACL configuration modes)

**Related Commands**
- `ip access-list` enters ACL configuration mode for editing IPv4 ACLs.
- `show ip access-lists` displays IPv4 and standard IPv4 ACLs.

**Examples**
- This command places the switch in std-ACL configuration mode to modify the `filter2` IPv4 ACL.
  ```
  switch(config)#ip access-list standard filter2  
  switch(config-std-acl-filter2)#
  ```
- This command saves changes to `filter2` ACL, then returns the switch to global configuration mode.
  ```
  switch(config-std-acl-filter2)#exit  
  switch(config)#
  ```
- This command discards changes to `filter2`, then returns the switch to global configuration mode.
  ```
  switch(config-std-acl-filter2)#abort  
  switch(config)#
  ```
**ip prefix-list**

The `ip prefix-list` command creates a prefix list or adds an entry to an existing list. Route map match commands use prefix lists to filter routes for redistribution into OSPF, RIP, or BGP domains.

A prefix list comprises all prefix list entries with the same label. The sequence numbers of the rules in a prefix list specify the order that the rules are applied to a route that the `match` command is evaluating.

The `no ip prefix-list` and `default ip prefix-list` commands delete the specified prefix list entry by removing the corresponding `ip prefix-list` statement from `running-config`. If the `no` or `default ip prefix-list` command does not list a sequence number, the command deletes all entries of the prefix list.

**Command Mode**  
Global Configuration

**Command Syntax**

```
    ip prefix-list list_name [SEQUENCE] FILTER_TYPE network_addr [MASK]
    no ip prefix-list list_name [SEQUENCE]
    default ip prefix-list list_name [SEQUENCE]
```

**Parameters**

- `list_name`  
The label that identifies the prefix list.
- `SEQUENCE`  
  Sequence number of the prefix list entry. Options include:
  - `<no parameter>`  
    entry’s number is ten plus highest sequence number in current list.
  - `seq seq_num`  
    number assigned to entry. Value ranges from 0 to 65535.
  - `FILTER_TYPE`  
    specifies route access when it matches IP prefix list. Options include:
  - `permit`  
    routes are permitted access when they match the specified subnet.
  - `deny`  
    routes are denied access when they match the specified subnet.
  - `network_addr`  
    Subnet upon which command filters routes. Format is CIDR or address-mask.
  - `MASK`  
    range of the prefix to be matched.
  - `<no parameter>`  
    exact match with the subnet mask is required.
  - `eq mask_e`  
    prefix length is equal to `mask_e`.
  - `ge mask_g`  
    range is from `mask_g` to 32.
  - `le mask_l`  
    range is from `subnet` mask length to `mask_l`.
  - `ge mask_l ge mask_g`  
    range is from `mask_g` to `mask_l`.
  - `mask_e, mask_l` and `mask_g` range from 1 to 32.

\[ \text{when le and ge are specified, subnet mask} > \text{mask_g} > \text{mask_l} \]

**Example**

- This command places the switch in IPv4 prefix list configuration mode to create an IPv4 prefix list named `route-one`.

```
switch(config)#ip prefix-list route-one
switch(config-ip-pfx)#
```
These commands create four different rules for the prefix-list named **route-one**.

```
switch(config)#ip prefix-list route-one
switch(config-ip-pfx)#seq 10 deny 10.1.1.0/24
switch(config-ip-pfx)#seq 20 deny 10.1.0.0/16
switch(config-ip-pfx)#seq 30 permit 12.15.4.9/32
switch(config-ip-pfx)#seq 40 deny 1.1.1.0/24
```
**ipv6 access-group**

The **ipv6 access-group** command applies an IPv6 or standard IPv6 access control list (ACL) to the configuration mode interface.

The **no ipv6 access-group** and **default ipv6 access-group** commands remove the corresponding **ipv6 access-group** command from *running-config*.

**Command Mode**
- Interface-Ethernet Configuration
- Interface-Port-Channel Configuration
- Interface-VLAN Configuration

**Command Syntax**

```
ipv6 access-group list_name DIRECTION
no ipv6 access-group list_name DIRECTION
default ipv6 access-group list_name DIRECTION
```

**Parameters**
- **list_name** name of ACL assigned to interface.
- **DIRECTION** transmission direction of packets, relative to interface. Valid options include:
  - **in** inbound packets.
  - **out** outbound packets.

**Examples**
- These commands assign the IPv6 ACL named **test2** to the Ethernet 3 interface.
  ```
  switch(config)#interface ethernet 3
  switch(config-if-Et3)#ipv6 access-group test2 in
  switch(config-if-Et3)#
  ```
**ipv6 access-group (Service ACLs)**

The `ipv6 access-group (Service ACLs)` command configures an IPv6 or standard IPv6 Service ACL to be applied by a control-plane service. The service is specified by the command mode in which the Service ACL is applied.

The `no ipv6 access-group (Service ACLs)` and `default ipv6 access-group (Service ACLs)` commands remove the corresponding `ipv6 access-group (Service ACLs)` command from `running-config`.

**Command Mode**
- Mgmt-SSH Configuration
- Mgmt-API Configuration
- Router-BGP Configuration
- Router-OSPF Configuration
- MPLS-LDP Configuration
- Queue-Monitor-Streaming Configuration
- MPLS-Ping Configuration
- Mgmt-Telnet Configuration

**Command Syntax**

```
ipv6 access-group ipv6_acl_name [vrf vrf_name] [in]
no ipv6 access-group [ipv6_acl_name] [vrf vrf_name] [in]
default ip access-group ipv6_acl_name [vrf vrf_name] [in]
```

**Parameters**

Parameters vary by process.

- `ipv6_acl_name` name of the IPv6 Service ACL assigned to control-plane service.
- `vrf vrf_name` specifies the VRF in which the Service ACL is to be applied.
- `in` specifies inbound connections or packets only (keyword required for SSH and Telnet services).

**Example**

These commands apply the IPv6 Service ACL `bgpacl` to the BGP routing protocol in VRF `purple`.

```
(config)# router bgp 5
(config-router-bgp)# vrf purple
(config-router-bgp-vrf-purple)# ipv6 access-group bgpacl
```

For additional configuration examples, see Section 24.3.2: Configuring Service ACLs and Displaying Status and Counters.
**ipv6 access-list**

The **ipv6 access-list** command places the switch in IPv6-ACL configuration mode, which is a group change mode that modifies an IPv6 access control list. The command specifies the name of the IPv6 ACL that subsequent commands modify and creates an ACL if it references a nonexistent list. All changes in a group change mode edit session are pending until the end of the session.

The **exit** command saves pending ACL changes to **running-config**, then returns the switch to global configuration mode. ACL changes are also saved by entering a different configuration mode.

The **abort** command discards pending ACL changes, returning the switch to global configuration mode.

The **no ipv6 access-list** and **default ipv6 access-list** commands delete the specified IPv6 ACL.

**Command Mode**

Global Configuration

**Command Syntax**

```
ipv6 access-list list_name
no ipv6 access-list list_name
default ipv6 access-list list_name
```

**Parameters**

- **list_name** Name of ACL. Must begin with an alphabetic character. Cannot contain spaces or quotation marks.

**Commands Available in IPv6-ACL configuration mode:**

- **deny** (IPv6 ACL)
- **no** <sequence number> (ACLs)
- **permit** (IPv6 ACL)
- **remark**
- **resequence** (ACLs)
- **show** (ACL configuration modes)

**Related Commands**

- **ipv6 access-list standard** enters std-ipv6-acl configuration mode for editing standard IPv6 ACLs.
- **show ipv6 access-lists** displays IPv6 and standard IPv6 ACLs.

**Examples**

- This command places the switch in IPv6-ACL configuration mode to modify the **filter1** IPv6 ACL.
  ```
  switch(config)#ipv6 access-list filter1
  switch(config-ipv6-acl-filter1)#
  ```

- This command saves changes to **filter1** ACL, then returns the switch to global configuration mode.
  ```
  switch(config-ipv6-acl-filter1)#exit
  switch(config)#
  ```

- This command discards changes to **filter1**, then returns the switch to global configuration mode.
  ```
  switch(config-ipv6-acl-filter1)#abort
  switch(config)#
  ```
ipv6 access-list standard

The **ipv6 access-list standard** command places the switch in std-IPv6-ACL-configuration mode, which is a group change mode that modifies a standard IPv6 access control list. The command specifies the name of the standard IPv6 ACL that subsequent commands modify and creates an ACL if it references a nonexistent list. All group change mode edit session changes are pending until the session ends.

The **exit** command saves pending ACL changes to **running-config**, then returns the switch to global configuration mode. Pending changes are also saved by entering a different configuration mode.

The **abort** command discards pending ACL changes, returning the switch to global configuration mode.

The **no ipv6 access-list standard** and **default ipv6 access-list standard** commands delete the specified ACL.

**Command Mode**
Global Configuration

**Command Syntax**

```
ipv6 access-list standard list_name
no ipv6 access-list standard list_name
default ipv6 access-list standard list_name
```

**Parameters**
- `list_name` Name of ACL.
  Must begin with an alphabetic character. Cannot contain spaces or quotation marks.

**Commands Available in std-IPv6-ACL configuration mode:**
- `deny` (Standard IPv6 ACL)
- `no <sequence number>` (ACLs)
- `permit` (Standard IPv6 ACL)
- `remark`
- `resequence` (ACLs)
- `show` (ACL configuration modes)

**Related Commands**
- **ipv6 access-list** enters IPv6-ACL configuration mode for editing IPv6 ACLs.
- **show ipv6 access-lists** displays IPv6 and standard IPv6 ACLs.

**Examples**
- This command places the switch in Std-IPv6 ACL configuration mode to modify the **filter2** ACL.
  ```
  switch(config)#ipv6 access-list standard filter2
  switch(config-std-ipv6-acl-filter2)#
  ```
- This command saves changes to **filter2** ACL, then returns the switch to global configuration mode.
  ```
  switch(config-std-ipv6-acl-filter2)#exit
  switch(config)#
  ```
- This command discards changes to **filter2**, then returns the switch to global configuration mode.
  ```
  switch(config-std-ipv6-acl-filter2)#abort
  switch(config)#
  ```
ipv6 prefix-list

The `ip prefix-list` command places the switch in IPv6 prefix-list configuration mode, which is a group change mode that modifies an IPv6 prefix list. The command specifies the name of the IPv6 prefix list that subsequent commands modify and creates a prefix list if it references a nonexistent list. All changes in a group change mode edit session are pending until the end of the session.

The `exit` command saves pending prefix list changes to `running-config`, then returns the switch to global configuration mode. ACL changes are also saved by entering a different configuration mode.

The `abort` command discards pending changes, returning the switch to global configuration mode.

The `no ipv6 prefix-list` and `default ipv6 prefix-list` commands delete the specified IPv6 prefix list.

Command Mode

Global Configuration

Command Syntax

```
ipv6 prefix-list  list_name
no ipv6 prefix-list list_name
default ipv6 prefix-list  list_name
```

Parameters

- `list_name` Name of prefix list.
  Must begin with an alphabetic character. Cannot contain spaces or quotation marks.

Commands Available in IPv6-pfx configuration mode:

- `deny (IPv6 Prefix List)`
- `permit (IPv6 Prefix List)`
- `seq (IPv6 Prefix Lists)`

Examples

- This command places the switch in IPv6 prefix-list configuration mode to modify the `route-five` prefix list.
  ```
  switch(config)#ipv6 prefix-list route-five
  switch(config-ipv6-pfx)#
  ```

- This command saves changes to the prefix list, then returns the switch to global configuration mode.
  ```
  switch(config-ipv6-pfx)#exit
  switch(config)#
  ```

- This command saves changes to the prefix list, then places the switch in interface-Ethernet mode.
  ```
  switch(config-ipv6-pfx)#interface ethernet 3
  switch(config-if-Et3)#
  ```

- This command discards changes to the prefix list, then returns the switch to global configuration mode.
  ```
  switch(config-ipv6-pfx)#abort
  switch(config)#
  ```
mac access-group

The **mac access-group** command applies a MAC access control list (MAC ACL) to the configuration mode interface.

The **no mac access-group** and **default mac access-group** commands remove the specified **mac access-group** command from **running-config**.

**Command Mode**

- Interface-Ethernet Configuration
- Interface-Port-Channel Configuration

**Command Syntax**

```
mac access-group list_name DIRECTION
no mac access-group list_name DIRECTION
default mac access-group list_name DIRECTION
```

**Parameters**

- `list_name` name of MAC ACL.
- `DIRECTION` transmission direction of packets, relative to interface. Valid options include:
  - `in` inbound packets.
  - `out` outbound packets.

**Restrictions**

Filtering of outbound packets by MAC ACLs is supported only on Helix, Trident, and Trident II platform switches.

**Example**

- These commands assign the MAC ACL named **mtest2** to the Ethernet 3 interface to filter inbound packets.

```
switch(config)#interface ethernet 3
switch(config-if-Et3)#mac access-group mtest2 in
switch(config-if-Et3)#
```
mac access-list

The `mac access-list` command places the switch in MAC-ACL configuration mode, which is a group change mode that modifies a MAC access control list. The command specifies the name of the MAC ACL that subsequent commands modify and creates an ACL if it references a nonexistent list. All changes in a group change mode edit session are pending until the end of the session.

The `exit` command saves pending ACL changes to `running-config`, then returns the switch to global configuration mode. ACL changes are also saved by entering a different configuration mode.

The `abort` command discards pending ACL changes, returning the switch to global configuration mode.

The `no mac access-list` and `default mac access-list` commands delete the specified list.

**Command Mode**

Global Configuration

**Command Syntax**

```markdown
mac access-list list_name
no mac access-list list_name
default mac access-list list_name
```

**Parameters**

- `list_name` Name of MAC ACL. Names must begin with an alphabetic character and cannot contain a space or quotation mark.

**Commands Available in MAC-ACL configuration mode:**

- `deny (MAC ACL)`
- `no <sequence number> (ACLs)`
- `permit (MAC ACL)`
- `remark`
- `resequence (ACLs)`
- `show (ACL configuration modes)`

**Examples**

- This command places the switch in MAC-ACL configuration mode to modify the `mfilter1` MAC ACL.
  ```bash
  switch(config)#mac access-list mfilter1
  switch(config-mac-acl-mfilter1)#
  ```
- This command saves changes to `mfilter1` ACL, then returns the switch to global configuration mode.
  ```bash
  switch(config-mac-acl-mfilter1)#exit
  switch(config)#
  ```
- This command saves changes to `mfilter1` ACL, then places the switch in interface-Ethernet mode.
  ```bash
  switch(config-mac-acl-mfilter1)#interface ethernet 3
  switch(config-if-Et3)#
  ```
- This command discards changes to `mfilter1`, then returns the switch to global configuration mode.
  ```bash
  switch(config-mac-acl-mfilter1)#abort
  switch(config)#
  ```
match (route-map)

The `match` command creates a route map statement entry that specifies one route filtering command. When a statement contains multiple match commands, the permit or deny filter applies to a route only if its properties are equal to corresponding parameters in each `match` command. When a route’s properties do not equal the command parameters, the route is evaluated against the next statement in the route map, as determined by sequence number. If all statements fail to permit or deny the route, the route is denied.

The `no match` and `default match` commands remove the `match` command from the configuration mode route map statement by deleting the corresponding command from `running-config`.

Note

The route map configuration supports only standard ACL.

Command Mode

Route-Map Configuration

Command Syntax

```
match CONDITION
no match CONDITION
default match CONDITION
```

Parameters

- **CONDITION** specifies criteria for evaluating a route. Options include:
  - as <1 to 4294967295> BGP Autonomous System number.
  - as-path path_name BGP Autonomous System path access list.
  - community NAME BGP community. Options for NAME include:
    - listname BGP community.
    - listname exact-match BGP community; list must match set that is present.
  - extcommunity listname BGP extended community. Options for NAME include:
    - listname BGP community.
    - listname exact-match BGP community; list must match set that is present.
  - interface INTF_NAME Specifies an interface. Options for INTF_NAME include:
    - ethernet e_num Ethernet interface.
    - loopback l_num Loopback interface.
    - port-channel p_num Port channel interface.
    - vlan v_num VLAN interface.
  - ip address LIST IPv4 address filtered by an ACL or prefix list. LIST options include:
    - access-list acl_name IPv4 address filtered by access control list (ACL).
    - prefix-list plv4_name IPv4 address filtered by IP prefix list.
  - ip next-hop prefix-list plv4_name IPv4 next-hop filtered by IP prefix list.
  - ip resolved-next-hop prefix-list plv4_name IPv4 resolved nexthop filtered by IP prefix list.
  - ipv6 address prefix-list plv6_name IPv6 address filtered by IPv6 prefix list.
  - ipv6 next-hop prefix-list plv6_name IPv6 next-hop filtered by IPv6 prefix list.
  - ipv6 resolved-next-hop prefix-list plv6_name IPv6 resolved nexthop filtered by IPv6 prefix list.
- **local-preference** `<1 to 4294967295>`  BGP local preference metric.
- **metric** `<1 to 4294967295>`  route metric.
- **metric-type** `OSPF_TYPE`  OSPF metric type. Options include:
  - **type-1**  OSPF type 1 metric.
  - **type-2**  OSPF type 2 metric.
- **source-protocol** `protocol_type`  Routing protocol of route’s source. Options include:
  - **bgp**
  - **connected**
  - **ospf**
  - **rip**
  - **static**
- **tag** `<1 to 4294967295>`  route tag.

**Related Commands**
- **route-map**

**Example**
- This command creates a route map match rule that filters routes from BGP AS 15.

```
switch(config)#route-map map1
switch(config-route-map-map1)#match as 15
switch(config-route-map-map1)#
```
no <sequence number> (ACLs)

The no <sequence number> command removes the rule with the specified sequence number from the ACL. The default <sequence number> command also removes the specified rule.

Command Mode

ACL Configuration
IPv6-ACL Configuration
Std-ACL Configuration
Std-IPv6-ACL Configuration
MAC-ACL Configuration

Command Syntax

   no line_num
   default line_num

Parameters

- line_num sequence number of rule to be deleted. Values range from 1 to 4294967295.

Example

- This command removes statement 30 from the list

  switch(config-acl-test1)#show
  IP Access List test1
  10 permit ip 10.10.10.0/24 any
  20 permit ip any host 10.20.10.1
  30 deny ip host 10.10.10.1 host 10.20.10.1
  40 permit ip any any
  50 remark end of list
  switch(config-acl-test1)#no 30
  switch(config-acl-test1)#show
  IP Access List test1
  10 permit ip 10.10.10.0/24 any
  20 permit ip any host 10.20.10.1
  40 permit ip any any
  50 remark end of list
permit (IPv4 ACL)

The `permit` command adds a permit rule to the configuration mode IPv4 access control list (ACL). Packets filtered by a permit rule are accepted by interfaces to which the ACL is applied. Sequence numbers determine rule placement in the ACL. Sequence numbers for commands without numbers are derived by adding 10 to the number of the ACL's last rule.

The `no permit` and `default permit` commands remove the specified rule from the configuration mode ACL. The `no <sequence number> (ACLs)` command also removes a specified rule from the ACL.

Command Mode

ACL Configuration

Command Syntax

```
[SEQ_NUM] permit PROTOCOL SOURCE_ADDR [SOURCE_PORT] DEST_ADDR [DEST_PORT] [FLAGS] [MESSAGE] [fragments] [tracked] [DSCP_FILTER] [TTL_FILTER] [log]
```

```
no permit PROTOCOL SOURCE_ADDR [SOURCE_PORT] DEST_ADDR [DEST_PORT] [FLAGS] [MESSAGE] [fragments] [tracked] [DSCP_FILTER] [TTL_FILTER] [log]
```

```
default permit PROTOCOL SOURCE_ADDR [SOURCE_PORT] DEST_ADDR [DEST_PORT] [FLAGS] [MESSAGE] [fragments] [tracked] [DSCP_FILTER] [TTL_FILTER] [log]
```

Commands use a subset of the listed fields. Available parameters depend on specified protocol. Use CLI syntax assistance to view options for specific protocols when creating a permit rule.

Parameters

- **SEQ_NUM**  
  Sequence number assigned to the rule. Options include:
  - `<no parameter>`  
  - `<1 – 4294967295>`  
    Number is derived by adding 10 to the number of the ACL’s last rule.
  - `<1 – 4294967295>`  
    Number assigned to entry.

- **PROTOCOL**  
  Protocol field filter. Values include:
  - `ahp`  
    Authentication Header Protocol (51).
  - `icmp`  
    Internet Control Message Protocol (1).
  - `igmp`  
    Internet Group Management Protocol (2).
  - `ip`  
    Internet Protocol v4 (4).
  - `ospf`  
    Open Shortest Path First (89).
  - `pim`  
    Protocol Independent Multicast (103).
  - `tcp`  
    Transmission Control Protocol (6).
  - `udp`  
    User datagram protocol (17).
  - `vrrp`  
    Virtual Router Redundancy Protocol (112).
  - `protocol_num`  
    Integer corresponding to an IP protocol. Values range from 0 to 255.

- **SOURCE_ADDR** and **DEST_ADDR**  
  Source and destination address filters. Options include:
  - `network_addr`  
    Subnet address (CIDR or address-mask).
  - `any`  
    Packets from all addresses are filtered.
  - `host ip_addr`  
    IP address (dotted decimal notation).
    Source and destination subnet addresses support discontiguous masks.

- **SOURCE_PORT** and **DEST_PORT**  
  Source and destination port filters. Options include:
• **any** all ports
• **eq port-1 port-2 ... port-n** A list of ports. Maximum list size is 10 ports.
• **neq port-1 port-2 ... port-n** The set of all ports not listed. Maximum list size is 10 ports.
• **gt port** The set of ports with larger numbers than the listed port.
• **lt port** The set of ports with smaller numbers than the listed port.
• **range port_1 port_2** The set of ports whose numbers are between the range.
• **fragments** filters packets with FO bit set (indicates a non-initial fragment packet).
• **FLAGS** flag bit filters (TCP packets). Use CLI syntax assistance (?) to display options.
• **MESSAGE** message type filters (ICMP packets). Use CLI syntax assistance (?) to display options.
• **tracked** rule filters packets in existing ICMP, UDP, or TCP connections.
  • Valid in ACLs applied to the control plane.
  • Validity in ACLs applied to data plane varies by switch platform.
• **DSCP_FILTER** rule filters packet by its DSCP value. Values include:
  • `<no parameter>` Rule does not use DSCP to filter packets.
  • **dscp dscp_value** Packets match if DSCP field in packet is equal to `dscp_value`.
• **TTL_FILTER** rule filters packet by its TTL (time-to-live) value. Values include:
  • **ttl eq ttl_value** Packets match if `ttl` in packet is equal to `ttl_value`.
  • **ttl gt ttl_value** Packets match if `ttl` in packet is greater than `ttl_value`.
  • **ttl lt ttl_value** Packets match if `ttl` in packet is less than `ttl_value`.
  • **ttl neq ttl_value** Packets match if `ttl` in packet is not equal to `ttl_value`.
  • Valid in ACLs applied to the control plane.
  • Validity in ACLs applied to data plane varies by switch platform.
• **log** triggers an informational log message to the console about the matching packet.
  • Valid in ACLs applied to the control plane.
  • Validity in ACLs applied to data plane varies by switch platform.

**Examples**
• This command appends a **permit** statement at the end of the ACL. The **permit** statement passes all OSPF packets from 10.10.1.1/24 to any host.
  
  switch(config)#ip access-list text1
  switch(config-acl-text1)#permit ospf 10.1.1.0/24 any
  switch(config-acl-text1)#

• This command inserts a **permit** statement with the sequence number 25. The **permit** statement passes all PIM packets through the interface.
  
  switch(config-acl-text1)#25 permit pim any any
  switch(config-acl-text1)#
**permit (IPv6 ACL)**

The `permit` command adds a permit rule to the configuration mode IPv6 access control list (ACL). Packets filtered by a permit rule are accepted by interfaces to which the ACL is applied. Sequence numbers determine rule placement in the ACL. Sequence numbers for commands without numbers are derived by adding 10 to the number of the ACL’s last rule.

The `no permit` and `default permit` commands remove the specified rule from the configuration mode ACL. The `no <sequence number> (ACLs)` command also removes a specified rule from the ACL.

**Command Mode**

IPv6-ACL Configuration

**Command Syntax**

```
[SEQ_NUM] permit PROT SRC_ADDR [SRC_PT] DEST_ADDR [DEST_PT] [FLAG] [MSG] [HOP] [tracked] [DSCP_FILTER] [log]
```

```
no permit PROT SRC_ADDR [SRC_PT] DEST_ADDR [DEST_PT] [FLAG] [MSG] [HOP] [tracked] [DSCP_FILTER] [log]
```

```
default permit PROT SRC_ADDR [SRC_PT] DEST_ADDR [DEST_PT] [FLAG] [MSG] [HOP] [tracked] [DSCP_FILTER] [log]
```

Commands use a subset of the listed fields. Available parameters depend on specified protocol. Use CLI syntax assistance to view options for specific protocols when creating a permit rule.

**Parameters**

- **SEQ_NUM**  
  Sequence number assigned to the rule. Options include:
  - `<no parameter>` Number is derived by adding 10 to the number of the ACL’s last rule.
  - `<1 – 4294967295>` Number assigned to entry.

- **PROT**  
  Protocol field filter. Values include:
  - `icmpv6` Internet Control Message Protocol for v6 (58).
  - `ipv6` Internet Protocol – IPv6 (41).
  - `ospf` Open Shortest Path First (89).
  - `tcp` Transmission Control Protocol (6).
  - `udp` User Datagram Protocol (17).
  - `protocol_num` integer corresponding to an IP protocol. Values range from 0 to 255.

- **SRC_ADDR** and **DEST_ADDR**  
  Source and destination address filters. Options include:
  - `ipv6_prefix` IPv6 address with prefix length (CIDR notation).
  - `any` Packets from all addresses are filtered.
  - `host ipv6_addr` IPv6 host address.

- **SRC_PT** and **DEST_PT**  
  Source and destination port filters. Options include:
  - `any` all ports.
  - `eq port-1 port-2 ... port-n` A list of ports. Maximum list size is 10 ports.
  - `neq port-1 port-2 ... port-n` The set of all ports not listed. Maximum list size is 10 ports.
  - `gt port` The set of ports with larger numbers than the listed port.
  - `lt port` The set of ports with smaller numbers than the listed port.
  - `range port_1 port_2` The set of ports whose numbers are between the range.
HOP filters by packet's hop-limit value. Options include:

- `<no parameter>` Rule does not use hop limit to filter packets.
- `hop-limit eq hop_value` Packets match if `hop-limit` value in packet equals `hop_value`.
- `hop-limit gt hop_value` Packets match if `hop-limit` in packet is greater than `hop_value`.
- `hop-limit lt hop_value` Packets match if `hop-limit` in packet is less than `hop_value`.
- `hop-limit neq hop_value` Packets match if `hop-limit` in packet is not equal to `hop_value`.

FLAG flag bit filters (TCP packets). Use CLI syntax assistance (?) to display options.

MSG message type filters (ICMPv6 packets). Use CLI syntax assistance (?) to display options.

tracked rule filters packets in existing ICMP, UDP, or TCP connections.

- Valid in ACLs applied to the control plane.
- Validity in ACLs applied to data plane varies by switch platform.

DSCP_FILTER rule filters packet by its DSCP value. Values include:

- `<no parameter>` Rule does not use DSCP to filter packets.
- `dscp dscp_value` Packets match if DSCP field in packet is equal to `dscp_value`.
- `log` triggers an informational log message to the console about the matching packet.

- Valid in ACLs applied to the control plane.
- Validity in ACLs applied to data plane varies by switch platform.

Example

This command appends a `permit` statement at the end of the ACL. The `permit` statement passes all IPv6 packets with the source address 3710:249a:c643:ef11::/64 and with any destination address.

```
switch(config)#ipv6 access-list text1
switch(config-acl-text1)#permit ipv6 3710:249a:c643:ef11::/64 any
switch(config-acl-text1)#
```
permit (IPv6 Prefix List)

The permit command adds a rule to the configuration mode IPv6 prefix list. Route map match commands use prefix lists to filter routes for redistribution into OSPF, RIP, or BGP domains. Routes are redistributed into the specified domain when they match the prefix that a permit statement specifies.

The no permit and default permit commands remove the specified rule from the configuration mode prefix list. The no seq (IPv6 Prefix Lists) command also removes the specified rule from the prefix list.

Command Mode
IPv6-pfx Configuration

Command Syntax

```
[SEQUENCE] deny ipv6_prefix [MASK]
```

Parameters
- **SEQUENCE** Sequence number assigned to the rule. Options include:
  - <no parameter> Number is derived by adding 10 to the number of the list's last rule.
  - `seq seq_num` Number is specified by `seq_num`. Value ranges from 0 to 65535.
- `ipv6_prefix` IPv6 prefix upon which command filters routes (CIDR notation).
- **MASK** Range of the prefix to be matched.
  - <no parameter> exact match with the subnet mask is required.
  - `eq mask_e` prefix length is equal to `mask_e`.
  - `ge mask_g` range is from `mask_g` to 128.
  - `le mask_l` range is from subnet mask length to `mask_l`.
  - `ge mask_l le mask_g` range is from `mask_g` to `mask_l`.
  - `mask_e, mask_l` and `mask_g` range from 1 to 128.

When `le` and `ge` are specified, the prefix list size > `mask_g > mask_l`

Example
- This command appends a permit statement at the end of the text1 prefix list. The permit statement allows redistribution of routes with the specified prefix.
  ```
  switch(config)#ipv6 prefix-list route-five
  switch(config-ipv6-pfx)#permit 3100::/64
  switch(config-ipv6-pfx)#
  ```
**permit (MAC ACL)**

The **permit** command adds a permit rule to the configuration mode MAC access control list packets through the interface to which the list is applied. Rule filters include protocol, source, and destination.

The **no permit** and **default permit** commands remove the specified rule from the configuration mode ACL. The **no <sequence number> (ACLs)** command also removes the specified rule from the ACL.

**Command Mode**

MAC-ACL Configuration

**Command Syntax**

```
[SEQ_NUM] permit SOURCE_ADDR DEST_ADDR [PROTOCOL] [log]
no permit SOURCE_ADDR DEST_ADDR [PROTOCOL] [log]
default permit SOURCE_ADDR DEST_ADDR [PROTOCOL] [log]
```

**Parameters**

- **SEQ_NUM**  Sequence number assigned to the rule. Options include:
  - `<no parameter>` Number is derived by adding 10 to the number of the ACL's last rule.
  - `<1 – 4294967295>` Number assigned to entry.
- **SOURCE_ADDR** and **DEST_ADDR**  source and destination address filters. Options include:
  - `mac_address mac_mask`  MAC address and mask
  - `any`  Packets from all addresses are filtered.
    `mac_address` specifies a MAC address in 3x4 dotted hexadecimal notation
    (hhhh.hhhh.hhhh)
    `mac_mask` specifies a MAC address mask in 3x4 dotted hexadecimal notation
    (hhhh.hhhh.hhhh)
  - `0` bits require an exact match to filter
  - `1` bits filter on any value
- **PROTOCOL**  protocol field filter. Values include:
  - `aarp`  Appletalk Address Resolution Protocol (0x80f3)
  - `appletalk`  Appletalk (0x809b)
  - `arp`  Address Resolution Protocol (0x806)
  - `ip`  Internet Protocol Version 4 (0x800)
  - `ipx`  Internet Packet Exchange (0x8137)
  - `lldp`  LLDP (0x88cc)
  - `novell`  Novell (0x8138)
  - `rarp`  Reverse Address Resolution Protocol (0x8035)
  - `protocol_num`  integer corresponding to a MAC protocol. Values range from 0 to 65535
- **log**  triggers an informational log message to the console about the matching packet.

**Examples**

- This command appends a **permit** statement at the end of the ACL. The **permit** statement passes all aarp packets from 10.1000.0000 through 10.1000.FFFF to any host.

  switch(config)#mac access-list text1
  switch(config-mac-acl-text1)#permit 10.1000.0000 0.0.FFFF any aarp
  switch(config-mac-acl-text1)#
• This command inserts a `permit` statement with the sequence number 25. The `permit` statement passes all packets through the interface.

  switch(config-mac-acl-text1)#25 permit any any
  switch(config-mac-acl-text1)#
permit (Standard IPv4 ACL)

The *permit* command adds a permit rule to the configuration mode standard IPv4 access control list (ACL). Standard ACL rules filter on the source field.

Packets filtered by a permit rule are accepted by interfaces to which the ACL is applied. Sequence numbers determine rule placement in the ACL. Sequence numbers for commands without numbers are derived by adding 10 to the number of the ACL’s last rule.

The *no permit* and *default permit* commands remove the specified rule from the configuration mode ACL. The *no <sequence number> (ACLs)* command also removes the specified rule from the ACL.

**Command Mode**
Std-ACL Configuration

**Command Syntax**

```
[SEQ_NUM] permit SOURCE_ADDR [log]
no permit SOURCE_ADDR [log]
default permit SOURCE_ADDR [log]
```

**Parameters**

- **SEQ_NUM** Sequence number assigned to the rule. Options include:
  - *<no parameter>* Number is derived by adding 10 to the number of the ACL’s last rule.
  - *<1 – 4294967295>* Number assigned to entry.
- **SOURCE_ADDR** source address filter. Options include:
  - *network_addr* subnet address (CIDR or address-mask).
  - *any* Packets from all addresses are filtered.
  - *host ip_addr* IP address (dotted decimal notation).

  Subnet addresses support discontiguous masks.
- **log** triggers an informational log message to the console about the matching packet.
  - Valid in ACLs applied to the control plane.
  - Validity in ACLs applied to data plane varies by switch platform.

**Example**

- This command appends a *permit* statement at the end of the ACL. The *permit* statement passes all packets with a source address of 10.10.1.1/24.

  ```
  switch(config)#ip access-list standard text1
  switch(config-standard-std-acl-text1)#permit 10.1.1.1/24
  switch(config-standard-std-acl-text1)#
  ```
permit (Standard IPv6 ACL)

The **permit** command adds a permit rule to the configuration mode standard IPv6 access control list. Standard ACL rules filter on the source field.

Packets filtered by a permit rule are accepted by interfaces to which the ACL is applied. Sequence numbers determine rule placement in the ACL. Sequence numbers for commands without numbers are derived by adding 10 to the number of the ACL’s last rule.

The **no permit** and **default permit** commands remove the specified rule from the configuration mode ACL. The **no <sequence number>** (ACLs) command also removes the specified rule from the ACL.

**Command Mode**
Std-IPv6-ACL Configuration

**Command Syntax**

```
[SEQ_NUM] permit SOURCE_ADDR
no permit SOURCE_ADDR
default permit SOURCE_ADDR
```

**Parameters**
- **SEQ_NUM** Sequence number assigned to the rule. Options include:
  - <no parameter> Number is derived by adding 10 to the number of the ACL’s last rule.
  - <1 – 4294967295> Number assigned to entry.
- **SOURCE_ADDR** source address filter. Options include:
  - *ipv6_prefix* IPv6 address with prefix length (CIDR notation).
  - *any* Packets from all addresses are filtered.
  - *host ipv6_addr* IPv6 host address.

**Example**
- This command appends a **permit** statement at the end of the ACL. The **permit** statement drops packets with a source address of 2103::/64.
  
  ```
  switch(config)#ipv6 access-list standard text1
  switch(config-std-acl-ipv6-text1)#permit 2103::/64
  ```

**remark**

The **remark** command adds a non-executable comment statement into the pending ACL. Remarks entered without a sequence number are appended to the end of the list. Remarks entered with a sequence number are inserted into the list as specified by the sequence number.

The **default remark** command removes the comment statement from the ACL.

The **no remark** command removes the comment statement from the ACL. The command can specify the remark by content or by sequence number.

**Command Mode**

- ACL Configuration
- IPv6-ACL Configuration
- Std-ACL Configuration
- Std-IPv6-ACL Configuration
- MAC-ACL Configuration

**Command Syntax**

```plaintext
remark text
line_num remark [text]
no remark text
default remark text
```

**Parameters**

- **text** the comment text.
- **line_num** sequence number assigned to the remark statement. Value ranges from 1 to 4294967295

**Example**

- This command appends a comment to the list.

```
switch(config-acl-test1)#remark end of list
switch(config-acl-test1)#show
IP Access List test1
  10 permit ip 10.10.10.0/24 any
  20 permit ip any host 10.20.10.1
  30 deny ip host 10.10.10.1 host 10.20.10.1
  40 permit ip any any
  50 remark end of list
```
resequence (ACLs)

The resequence command assigns sequence numbers to rules in the configuration mode ACL. Command parameters specify the number of the first rule and the numeric interval between consecutive rules.

Maximum rule sequence number is 4294967295.

Command Mode

ACL Configuration
IPv6-ACL Configuration
Std-ACL Configuration
Std-IPv6-ACL Configuration
MAC-ACL Configuration

Command Syntax

resequence [start_num [inc_num]]

Parameters

- **start_num** sequence number assigned to the first rule. Default is 10.
- **inc_num** numeric interval between consecutive rules. Default is 10.

Example

- The resequence command rennumbers the list, starting the first command at number 100 and incrementing subsequent lines by 20.

  ```
  switch(config-acl-test1)#show
  IP Access List test1
  10 permit ip 10.10.10.0/24 any
  20 permit ip any host 10.20.10.1
  30 deny ip host 10.10.10.1 host 10.20.10.1
  40 permit ip any any
  50 remark end of list
  switch(config-acl-test1)#resequence 100 20
  switch(config-acl-test1)#show
  IP Access List test1
  100 permit ip 10.10.10.0/24 any
  120 permit ip any host 10.20.10.1
  140 deny ip host 10.10.10.1 host 10.20.10.1
  160 permit ip any any
  180 remark end of list
  ```
route-map

The `route-map` command places the switch in route map configuration mode, which is a group change mode that modifies a route map statement. The command specifies the name and number of the route map statement that subsequent commands modify and creates a route map statement if it references a nonexistent statement. All changes in a group change mode edit session are pending until the end of the session.

Route maps define commands for redistributing routes between routing protocols. A route map statement is identified by a name, filter type (permit or deny), and sequence number. Statements with the same name are components of a single route map; the sequence number determines the order in which the statements are compared to a route.

The `exit` command saves pending route map statement changes to `running-config`, then returns the switch to global configuration mode. ACL changes are also saved by entering a different configuration mode.

The `abort` command discards pending changes, returning the switch to global configuration mode.

The `no route-map` and `default route-map` commands delete the specified route map statement from `running-config`.

**Note**
The route map configuration supports only standard ACL.

### Command Mode
Global Configuration

### Command Syntax

```plaintext
route-map  map_name  [FILTER_TYPE]  [sequence_number]
no route-map  map_name  [FILTER_TYPE]  [sequence_number]
default route-map  map_name  [FILTER_TYPE]  [sequence_number]
```

### Parameters
- **map_name** label assigned to route map. Protocols reference this label to access the route map.
- **FILTER_TYPE** disposition of routes matching commands specified by route map statement.
  - `permit` routes are redistributed when they match route map statement.
  - `deny` routes are not redistributed when they match route map statement.
  - `<No parameter>` assigns `permit` as the `FILTER_TYPE`.

When a route does not match the route map criteria, the next statement within the route map is evaluated to determine the redistribution action for the route.

- **sequence_number** the route map position relative to other statements with the same name.
  - `<no parameter>` sequence number of 10 (default) is assigned to the route map.
  - `<1-16777215>` specifies sequence number assigned to route map.

### Commands Available in route map configuration mode:
- `continue (route map)`
- `match (route-map)`
- `set (route-map)`
Examples

- This command creates the route map named `map1` and places the switch in route map configuration mode. The route map is configured as a permit map.
  
  ```
  switch(config)#route-map map1 permit 20
  switch(config-route-map-map1)#
  ```

- This command saves changes to `map1` route map, then returns the switch to global configuration mode.
  
  ```
  switch(config-route-map-map1)#exit
  switch(config)#
  ```

- This command saves changes to `map1` route map, then places the switch in interface-Ethernet mode.
  
  ```
  switch(config-route-map-map1)#interface ethernet 3
  switch(config-if-Et3)#
  ```

- This command discards changes to `map1` route map, then returns the switch to global configuration mode.
  
  ```
  switch(config-route-map-map1)#abort
  switch(config)#
  ```
seq (IPv6 Prefix Lists)

The `no seq` command removes the rule with the specified sequence number from the ACL. The `default seq` command also removes the specified rule.

The `seq` keyword is a command option used at the beginning of `deny (IPv6 Prefix List)` and `permit (IPv6 Prefix List)` commands that places a new rule between two existing rules.

**Command Mode**

IPv6-pfx Configuration

**Command Syntax**

```
no seq  line_num
default seq line_num
```

**Parameters**

- `line_num` sequence number of rule to be deleted. Valid rule numbers range from 0 to 65535.

**Example**

- These commands remove rule 20 from the `map1` prefix list, then displays the resultant list.

  ```
  switch(config)#ipv6 prefix-list map1
  switch(config-ipv6-pfx)#no seq 20
  switch(config-ipv6-pfx)#exit
  switch(config)#show ipv6 prefix-list map1
  ipv6 prefix-list map1
  seq 10 permit 3:4e96:8ca1:33cf::/64
  seq 15 deny 3:4400::/64
  seq 30 permit 3:1bca:3ff2:634a::/64
  seq 40 permit 3:1bca:1141:ab34::/64
  switch(config)#
  ```
set (route-map)

The `set` command specifies modifications to routes that are selected for redistribution by the configuration mode route map.

The `no set` and `default set` commands remove the specified `set` command from the configuration mode route map statement by deleting the corresponding `set` command from `running-config`.

**Command Mode**

Route-Map Configuration

**Command Syntax**

```
set CONDITION
no set CONDITION
default set CONDITION
```

**Parameters**

- `CONDITION` specifies the route modification parameter and value. Options include:
  - `as-path prepend` BGP AS number that is prepended to as-path. For details, see the `set as-path prepend` command.
    - `<1 to 4294967295>` BGP AS number to prepend.
    - `auto` use peer AS number for inbound and local AS for outbound to prepend.
  - `distance <1 to 255>` Protocol independent administrative distance.
  - `ip next-hop ipv4_address` next hop IPv4 address.
  - `ip next-hop peer-address` Use BGP peering address as next hop IPv4 address.
  - `ipv6 next-hop ipv6_address` next hop IPv6 address.
  - `ipv6 next-hop peer-address` Use BGP peering address as next hop IPv6 address.
  - `local-preference <1 to 4294967295>` BGP local preference metric.
  - `metric <1 to 4294967295>` route metric.
  - `metric +<1 to 4294967295>` add specified value to current route metric.
  - `metric – <1 to 4294967295>` subtract specified value to current route metric.
  - `metric-type OSPF_TYPE` OSPF metric type. Options include:
    - `type-1` OSPF type 1 metric.
    - `type-2` OSPF type 2 metric.
  - `origin O_TYPE` BGP origin attribute. Options for `O_TYPE` include
    - `egp` exterior BGP route.
    - `igp` interior BGP route.
    - `incomplete` BGP route of unknown origin.
  - `tag <1 to 4294967295>` route tag.
  - `weight <1 to 65535>` BGP weight parameter.

**Related Commands**

- `route-map` enters route-map configuration mode.
- `set (route-map)` specifies community modifications for the redistributed routes.
- `set community (route-map)` specifies extended community modifications for the redistributed routes.
Example

- This command creates a route map entry that sets the local preference metric to 100 on redistributed routes.

```plaintext
switch(config)#route-map map1
switch(config-route-map-map1)#set local-preference 100
switch(config-route-map-map1)#
```
set as-path match

The **set as-path match** command configures the AS_PATH attribute for prefixes that are either received from a BGP neighbor or advertised to a BGP neighbor in the route map configuration mode.

The **no set as-path match** command removes the AS path specified for the BGP prefix.

**Command Mode**

Route-Map Configuration

**Command Syntax**

```
set as-path match all replacement [none | auto] as_path
no set as-path match all replacement [none | auto] as_path
```

**Parameters**

- **none** replaces the AS-Path of the matching routes with a null or an empty AS-Path.
- **auto** if the specific route map is applied as an inbound policy to a corresponding BGP neighbor statement, then replace the AS_PATH of the prefixes received from this neighbor with the neighbor’s AS number. If this route map is applied as an outbound policy to a corresponding neighbor statement, then replace the AS_PATH of the prefixes advertised to this neighbor with the locally configured AS number.
- **as_path** replaces the AS-Path of the matching routes with an arbitrary AS-Path.
Example

- This command replaces the AS-Path with the “none" option.

```plaintext
switch#show ip bgp neighbors 80.80.1.2 advertised-routes
BGP routing table information for VRF default
Router identifier 202.202.1.1, local AS number 200
Route status codes: s - suppressed, * - valid, > - active, # - not installed, E - ECMP head, e - ECMP
S - Stale, c - Contributing to ECMP, b - backup, L - labeled-unicast, q - Queued for advertisement
Origin codes: i - IGP, e - EGP, ? - incomplete
AS Path Attributes: Or-ID - Originator ID, C-LST - Cluster List, LL Nexthop - Link Local Nexthop

Network Next Hop Metric LocPref Weight Path
* > 101.101.1.0/24 80.80.1.1 - - - 200 i
* > 102.102.1.0/24 80.80.1.1 - - - 200 i
* > 103.103.1.0/24 80.80.1.1 - - - 200 302 i
* > 202.202.1.0/24 80.80.1.1 - - - 200 i
switch#configuration terminal
switch(config)#route-map foo permit 10
switch(config-route-map-foo)#set as-path match all replacement none
switch(config-route-map-foo)#exit
switch(config)#router bgp 200
switch(config-router-bgp)#neighbor 80.80.1.2 route-map foo out
switch(config-router-bgp)#end

switch#show ip bgp neighbors 80.80.1.2 advertised-routes
BGP routing table information for VRF default
Router identifier 202.202.1.1, local AS number 200
Route status codes: s - suppressed, * - valid, > - active, # - not installed, E - ECMP head, e - ECMP
S - Stale, c - Contributing to ECMP, b - backup, L - labeled-unicast, q - Queued for advertisement
Origin codes: i - IGP, e - EGP, ? - incomplete
AS Path Attributes: Or-ID - Originator ID, C-LST - Cluster List, LL Nexthop - Link Local Nexthop

Network Next Hop Metric LocPref Weight Path
* > 101.101.1.0/24 80.80.1.1 - - - 200 i
* > 102.102.1.0/24 80.80.1.1 - - - 200 i
* > 103.103.1.0/24 80.80.1.1 - - - 200 i
* > 202.202.1.0/24 80.80.1.1 - - - 200 i
```

The AS-Path of matching prefixes are replaced with an empty or a null AS-Path. AS 302 is removed from prefix 103.103.1.0/24 as shown in the above output.
- This command replaces the AS-Path with the “auto” option.

```plaintext
switch(config)#route-map foo permit 10
switch(config-route-map-foo)#set as-path match all replacement auto
switch(config-route-map-foo)#end
switch#show ip bgp neighbors 80.80.1.2 advertised-routes
BGP routing table information for VRF default
Router identifier 202.202.1.1, local AS number 200
Route status codes: s - suppressed, * - valid, > - active, # - not installed, E - ECMP head, e - ECMP
S - Stale, c - Contributing to ECMP, b - backup, L - labeled-unicast, q - Queued for advertisement
Origin codes: i - IGP, e - EGP, ? - incomplete
AS Path Attributes: Or-ID - Originator ID, C-LST - Cluster List, LL Nexthop - Link Local Nexthop

Network Next Hop Metric LocPref Weight Path
* > 101.101.1.0/24 80.80.1.1 - - - 200 200 i
* > 102.102.1.0/24 80.80.1.1 - - - 200 200 i
* > 103.103.1.0/24 80.80.1.1 - - - 200 200 i
* > 202.202.1.0/24 80.80.1.1 - - - 200 200 i

The AS-Path of matching prefixes are replaced with the locally configured AS – 200.
```

- This command replaces the AS-Path with another AS-Path.

```plaintext
switch(config)#route-map foo permit 10
switch(config-route-map-foo)#set as-path match all replacement 500 600
switch(config-route-map-foo)#end
switch#show ip bgp neighbors 80.80.1.2 advertised-routes
BGP routing table information for VRF default
Router identifier 202.202.1.1, local AS number 200
Route status codes: s - suppressed, * - valid, > - active, # - not installed, E - ECMP head, e - ECMP
S - Stale, c - Contributing to ECMP, b - backup, L - labeled-unicast, q - Queued for advertisement
Origin codes: i - IGP, e - EGP, ? - incomplete
AS Path Attributes: Or-ID - Originator ID, C-LST - Cluster List, LL Nexthop - Link Local Nexthop

Network Next Hop Metric LocPref Weight Path
* > 101.101.1.0/24 80.80.1.1 - - - 200 500 600 i
* > 102.102.1.0/24 80.80.1.1 - - - 200 500 600 i
* > 103.103.1.0/24 80.80.1.1 - - - 200 500 600 i
* > 202.202.1.0/24 80.80.1.1 - - - 200 500 600 i

The AS-Path of matching prefixes are replaced with 500 600 as configured.
```
This command replaces the AS-Path with a combination of ‘auto’ and an AS-Path.

switch(config)#route-map foo permit 10
switch(config-route-map-foo)#set as-path match all replacement auto 500 600
switch(config-route-map-foo)#end

switch#show ip bgp neighbors 80.80.1.2 advertised-routes

BGP routing table information for VRF default
Router identifier 202.202.1.1, local AS number 200
Route status codes: s - suppressed, * - valid, > - active, # - not installed, E - ECMP head, e - ECMP
S - Stale, c - Contributing to ECMP, b - backup, L - labeled-unicast, q - Queued for advertisement
Origin codes: i - IGP, e - EGP, ? - incomplete
AS Path Attributes: Or-ID - Originator ID, C-LST - Cluster List, LL Nexthop - Link Local Nexthop

<table>
<thead>
<tr>
<th>Network</th>
<th>Next Hop</th>
<th>Metric</th>
<th>LocPref</th>
<th>Weight</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>* &gt; 101.101.1.0/24 80.80.1.1</td>
<td>- -</td>
<td>- -</td>
<td>200 200</td>
<td>500 600</td>
<td>i</td>
</tr>
<tr>
<td>* &gt; 102.102.1.0/24 80.80.1.1</td>
<td>- -</td>
<td>- -</td>
<td>200 200</td>
<td>500 600</td>
<td>i</td>
</tr>
<tr>
<td>* &gt; 103.103.1.0/24 80.80.1.1</td>
<td>- -</td>
<td>- -</td>
<td>200 200</td>
<td>500 600</td>
<td>i</td>
</tr>
<tr>
<td>* &gt; 202.202.1.0/24 80.80.1.1</td>
<td>- -</td>
<td>- -</td>
<td>200 200</td>
<td>500 600</td>
<td>i</td>
</tr>
</tbody>
</table>

The AS-Path of matching prefixes are replaced with the locally configured AS – 200 and 500 600.
The `set as-path prepend` command adds a `set` statement to a route map to prepend one or more autonomous system (AS) numbers to the AS PATH attribute of a BGP route.

The `no set as-path prepend` and `default set as-path prepend` commands remove the specified set statements from the route map and update all corresponding routes.

**Command Mode**

Route-Map Configuration

**Command Syntax**

```
set as-path prepend {auto | as_number ... [auto | as_number]} | last-as count
no set as-path prepend {auto | as_number ... [auto | as_number]} | last-as count
default set as-path prepend {auto | as_number ... [auto | as_number]} | last-as count
```

**Parameters**

- `auto` prepends the peer AS number for peer inbound route maps and the local AS number for peer outbound route maps.
- `as_number` prepends the specified AS number. Can be entered in plain notation (values range from 1-4294967295) or in asdot notation as described in RFC 5396. In asdot notation, AS numbers from 1-65535 are entered in plain notation, and AS numbers from 65536 to 4294967295 are entered as two values separated by a dot. The first value is high-order and represents a multiple of 65536; the second value is low-order and represents a decimal integer. For example, AS number 65552 can be entered as either 65552 or 1.16 (i.e., 1*65536+16). However they are entered, AS numbers are stored internally in plain decimal notation and will appear that way in `show` outputs.
- `last-as count` prepends the last AS number in the AS path `count` times. Values range from 1 to 15. This is mutually exclusive with the use of the `auto` keyword or the entry of one or more specified AS numbers, and is not supported in multi-agent mode.

**Examples**

- These commands create a route-map entry that prepends AS number 64496 and prepends either the peer or local AS number twice.

```
switch(config)#route-map map1
switch(config-route-map-map1)#set as-path prepend 64496 auto auto
switch(config-route-map-map1)#exit
switch(config)#show route-map map1
route-map map1 permit 10
  Description:
  Match clauses:
  SubRouteMap:
  Set clauses:
    set as-path prepend 64496 auto auto
switch(config)#
```
The commands create a route-map entry that prepends AS numbers 64496, 64498, and 65552.

```plaintext
switch(config)#route-map map2
switch(config-route-map-map2)#set as-path prepend 64496 64498 65552
switch(config-route-map-map2)#exit
switch(config)#show route-map map2
route-map map2 permit 10
  Description:
  Match clauses:
  SubRouteMap:
  Set clauses:
    set as-path prepend 64496 64498 65552
switch(config)#
```

These commands create a route map entry that prepends the last AS number 12 times.

```plaintext
switch(config)#route-map map3
switch(config-route-map-map3)#set as-path prepend last-as 12
switch(config-route-map-map3)#exit
switch(config)#show route-map map3
route-map map3 permit 10
  Description:
  Match clauses:
  SubRouteMap:
  Set clauses:
    set as-path prepend last-as 12
switch(config)#
```
set community (route-map)

The set community command specifies community attribute modifications to routes that are selected for redistribution by the configuration mode route map. The set community none command removes community attributes from the route.

The no set community and default set community commands remove the specified community from the configuration mode route map statement by deleting the corresponding statement from the running configuration.

Command Mode
Route-Map Configuration

Command Syntax

```
set community [GSHUT | aa:nn | community-list | internet | local-as | no-advertise | no-export | none | number]
no set community [GSHUT | aa:nn | additive | community-list | delete | internet | local-as | no-advertise | no-export | none | number]
default set community [GSHUT | aa:nn | additive | community-list | delete | internet | local-as | no-advertise | no-export | none | number]
```

Parameters
- **GSHUT** configures a graceful shutdown in BGP.
- **aa:nn** configures the community AS and network number, separated by colon. Value ranges from 0:0 to 65535:65535.
- **community-list** A label for community list.
- **internet** advertises route to the Internet community.
- **local-as** advertises route only to local peers.
- **no-advertise** does not advertise route to any peer.
- **no-export** advertises route only within BGP AS boundary.
- **none** does not provide any community attributes.
- **number** configures the community number. Value ranges from 1 to 4294967040.
- **additive** adds specified attributes to the current community.
- **delete** removes specified attributes from the current community.

Related Commands
- ip community-list
- route-map
- set (route-map)
- set community (route-map)
Guideline
EOS does not support disabling the process of graceful shutdown community.

Example
- This command advertises routes only to local peers.
  ```
  switch(config-route-map-map1)#show active
  route-map map1 permit 10
          match community instances <= 50
          set community 0:456 0:2345
  switch(config-route-map-map1)#set community local-as
  switch(config)#route-map map1
  switch(config-route-map-map1)#ip community-list 345 permit 23
  switch(config)#route-map map1
  switch(config-route-map-map1)#show active
  route-map map1 permit 10
          match community instances <= 50
          set community 0:456 0:2345 local-as
  switch(config-route-map-map1)#
  ```
set extcommunity (route-map)

The `set extcommunity` command specifies extended community attribute modifications to routes that are selected for redistribution by the configuration mode route map. The `set extcommunity none` command removes extended community attributes from the route.

The `no set extcommunity` and `default set extcommunity` commands remove the specified `set extcommunity` command from the configuration mode route map statement by deleting the corresponding statement from `running-config`.

Command Mode

Route-Map Configuration

Command Syntax

```
set extcommunity COND_1 [COND_2] [COND_N] [MOD_TYPE]
set extcommunity none
no set extcommunity COND_1 [COND_2] [COND_N] [MOD_TYPE]
no set extcommunity none
default set extcommunity COND_1 [COND_2] [COND_N] [MOD_TYPE]
default set extcommunity none
```

Parameters

- **COND_X** Specifies extended community route map modification. Command may contain multiple attributes. Options include:
  - `rt ASN:nn` Route target attribute (AS:network number).
  - `rt IP-address:nn` Route target attribute (IP address: network number).
  - `soo ASN:nn` Site of origin attribute (AS:network number).
  - `soo IP-address:nn` Site of origin attribute (IP address: network number).

- **MOD_TYPE** Specifies route map modification method. Options include:
  - `<no parameter>` command replaces existing route map with specified parameters.
  - `additive` command adds specified parameters to existing route map.
  - `delete` command removes specified parameters from existing route map.

Related Commands

- `route-map` enters route map configuration mode.
- `set (route-map)` specifies attribute modifications for the redistributed routes
- `set (route-map)` specifies community modifications for the redistributed routes.

Example

- This command creates a route map entry in map1 that sets the route target extended community attribute.

```
switch(config)#route-map map1
switch(config-route-map-map1)#set extcommunity rt 10.13.2.4:100
switch(config-route-map-map1)#
```
show (ACL configuration modes)

The `show` command displays the contents of an access control list (ACL).

- **show** or **show pending** – displays the list as modified in ACL configuration mode.
- **show active** – displays the list as stored in `running-config`.
- **show comment** – displays the comment stored with the list.
- **show diff** – displays the modified and stored lists, with flags denoting the modified rules.

Exiting the ACL configuration mode stores all pending ACL changes to `running-config`.

**Command Mode**

ACL Configuration
IPv6-ACL Configuration
Std-ACL Configuration
Std-IPv6-ACL Configuration
MAC-ACL Configuration

**Command Syntax**

```
show
show active
show comment
show diff
show pending
```

**Examples**

The examples in this section assume these ACL commands are entered as specified.

These commands are stored in running-config:

```
10 permit ip 10.10.10.0/24 any
20 permit ip any host 10.21.10.1
30 deny ip host 10.10.10.1 host 10.20.10.1
40 permit ip any any
50 remark end of list
```

The current edit session removed this command. This change is not yet stored to running-config:

```
20 permit ip any host 10.21.10.1
```

The current edit session added these commands ACL. They are not yet stored to running-config:

```
20 permit ip 10.10.0.0/16 any
25 permit tcp 10.10.20.0/24 any
45 deny pim 239.24.124.0/24 10.5.8.4/30
```

- **This command displays the ACL, as stored in the configuration.**

```
switch(config-acl-test_1)#show active
IP Access List test_1
  10 permit ip 10.10.10.0/24 any
  20 permit ip any host 10.21.10.1
  30 deny ip host 10.10.10.1 host 10.20.10.1
  40 permit ip any any
  50 remark end of list
```
This command displays the pending ACL, as modified in ACL configuration mode.

```
switch(config-acl-test_1)#show pending
IP Access List test_1
    10 permit ip 10.10.10.0/24 any
    20 permit ip 10.10.0.0/16 any
    25 permit tcp 10.10.20.0/24 any
    30 deny ip host 10.10.10.1 host 10.20.10.1
    40 permit ip any any
    45 deny pim 239.24.124.0/24 10.5.8.4/30
    50 remark end of list
```

This command displays the difference between the saved and modified ACLs.

- Rules added to the pending list are denoted with a plus sign (+).
- Rules removed from the saved list are denoted with a minus sign (-)

```
switch(config-acl-test_1)#show diff
---
+++ @@ -1,7 +1,9 @@
IP Access List test_1
    10 permit ip 10.10.10.0/24 any
    20 permit ip any host 10.21.10.1
+   20 permit ip 10.10.0.0/16 any
+   25 permit tcp 10.10.20.0/24 any
    30 deny ip host 10.10.10.1 host 10.20.10.1
    40 permit ip any any
+   45 deny pim 239.24.124.0/24 10.5.8.4/30
```
**show ip access-lists**

The `show ip access-list` command displays the contents of IPv4 and standard IPv4 access control lists (ACLs) on the switch. Use the `summary` option to display only the name of the lists and the number of lines in each list.

**Command Mode**

Privileged EXEC

**Command Syntax**

```
show ip access-list [LIST] [SCOPE]
```

**Parameters**

- **LIST** name of lists to be displayed. Selection options include:
  - `<no parameter>` all IPv4 ACLs are displayed.
  - `list_name` specified IPv4 ACL is displayed.
- **SCOPE** information displayed. Selection options include:
  - `<no parameter>` all rules in the specified lists are displayed.
  - `summary` the number of rules in the specified lists are displayed.

**Examples**

- **This command displays all rules in test1 IPv4 ACL.**
  
  switch#show ip access-list list2
  IP Access List list2
  10 permit ip 10.10.10.0/24 any
  20 permit ip any host 10.20.10.1
  30 deny ip host 10.10.10.1 host 10.20.10.1
  switch#

- **This command displays the name of, and number of rules in, each list on the switch.**

  switch#show ip access-list summary
  IPV4 ACL default-control-plane-acl
  Total rules configured: 12
  Configured on: control-plane
  Active on : control-plane

  IPV4 ACL list2
  Total rules configured: 3

  IPV4 ACL test1
  Total rules configured: 6

  Standard IPV4 ACL test_1
  Total rules configured: 1

  IPV4 ACL test_3
  Total rules configured: 0

  switch#
This command displays the summary and lists all the configured IPv4 ACLs.

```
switch #show ip access-lists summary
IPV4 ACL default-control-plane-acl [readonly]
    Total rules configured: 17
    Configured on Ingress: control-plane(default VRF)
    Active on Ingress: control-plane(default VRF)

IPV4 ACL ipAclLimitTest
    Total rules configured: 0
    Configured on Egress: V12148,2700
    Active on Egress: V12148,2700
```
show ip prefix-list

The **show ip prefix-list** command displays all rules for the specified IPv4 prefix list. The command displays all IPv4 prefix list rules if a prefix list name is not specified.

**Command Mode**

`EXEC`

**Command Syntax**

```
show ip prefix-list [DISPLAY_ITEMS]
```

**Parameters**

- **DISPLAY_ITEMS** specifies the name of prefix lists for which rules are displayed. Options include:
  - `<no parameter>` all IPv4 prefix list rules are displayed.
  - `list_name` specifies the IPv4 prefix list for which rules are displayed.

**Example**

- This command displays all rules in the route-one IPv4 prefix list:
  ```
  switch(config-ip-pfx)#show ip prefix-list
  ip prefix-list route-one
      seq 10 deny 10.1.1.0/24
      seq 20 deny 10.1.0.0/16
      seq 30 permit 12.15.4.9/32
      seq 40 deny 1.1.1.0/24
  switch(config-ip-pfx)#
  ```
**show ipv6 access-lists**

The `show ipv6 access-list` command displays the contents of all IPv6 access control lists (ACLs) on the switch. Use the `summary` option to display only the name of the lists and the number of lines in each list.

**Command Mode**

Privileged EXEC

**Command Syntax**

```
show ipv6 access-list [LIST] [SCOPE]
```

**Parameters**

- **LIST** name of lists to be displayed. Selection options include:
  - `<no parameter>` all IPv6 ACLs are displayed.
  - `list_name` specified IPv6 ACL is displayed.
- **SCOPE** information displayed. Selection options include:
  - `<no parameter>` all rules in the specified lists are displayed.
  - `summary` the number of rules in the specified lists are displayed.

**Examples**

- This command displays all rules in test1 IPv6 ACL.
  ```
  switch#show ipv6 access-list list2
  IP Access List list2
  10 permit ipv6 3891:3c58:6300::/64 any
  20 permit ipv6 any host 2fe1:b468:024a::
  30 deny ipv6 host 3411:91c1:: host 4210:cc23:d2de::
  switch#
  ```

- This command displays the name of, and number of rules in, each list on the switch.
  ```
  switch#show ipv6 access-list summary
  IPV6 ACL list2
  Total rules configured: 3
  
  IPV6 ACL test1
  Total rules configured: 6
  
  IPV6 ACL test_1
  Total rules configured: 1
  
  Standard IPV6 ACL test_3
  Total rules configured: 0
  ```

  switch#
show ipv6 prefix-list

The `show ipv6 prefix-list` command displays all rules for the specified IPv6 prefix list. The command displays all IPv6 prefix lists if a prefix list name is not specified.

**Command Mode**

EXEC

**Command Syntax**

`show ipv6 prefix-list [DISPLAY_ITEMS]`

**Parameters**

- `DISPLAY_ITEMS` specifies the name of prefix lists for which rules are displayed. Options include:
  - `<no parameter>` all IPv6 prefix lists are displayed.
  - `list_name` specifies the IPv6 prefix list for which rules are displayed.

**Examples**

- This command displays all rules in the map1 IPv6 prefix list:

  ```
  switch>show ipv6 prefix-list map1
  ipv6 prefix-list map1
  seq 10 permit 3:4e96:8ca1:33cf::/64
  seq 15 deny 3:4400::/64
  seq 20 permit 3:11b1:8fe4:1aac::/64
  seq 30 permit 3:1bca:3ff2:634a::/64
  seq 40 permit 3:1bca:1141:ab34::/64
  switch>
  ```

- This command displays all prefix lists:

  ```
  switch>show ipv6 prefix-list
  ipv6 prefix-list map1
  seq 10 permit 3:4e96:8ca1:33cf::/64
  seq 15 deny 3:4400::/64
  seq 20 permit 3:11b1:8fe4:1aac::/64
  seq 30 permit 3:1bca:3ff2:634a::/64
  seq 40 permit 3:1bca:1141:ab34::/64
  ipv6 prefix-list FREDD
  ipv6 prefix-list route-five
  ipv6 prefix-list map2
  seq 10 deny 10:1:1:1::/64 ge 72 le 80
  seq 20 deny 10:1::/32
  switch>
  ```
show mac access-lists

The `show mac access-list` command displays the contents of all MAC access control lists (ACLs) on the switch. Use the summary to display only the name of the lists and the number of lines in each list.

Command Mode
Privileged EXEC

Command Syntax
```
show mac access-lists [LIST] [SCOPE]
```

Parameters
- **LIST** name of lists to be displayed. Selection options include:
  - `<no parameter>` command displays all ACLs.
  - `list_name` command displays ACL specified by parameter.
- **SCOPE** information displayed. Selection options include:
  - `<no parameter>` command displays all rules in specified lists.
  - `summary` command displays the number of rules in specified lists.

Examples
- This command displays all rules in `mtest2` MAC ACL.
  
  ```
  switch#show mac access-list mlist2
  IP Access List mlist2
  10 permit 1024.4510.F125 0.0.0 any aarp
  20 permit any 4100.4500.0000 0.FF.FFFF novell
  30 deny any any
  switch#
  ```

- This command displays the number of rules in each MAC ACL on the switch.
  
  ```
  switch#show mac access-list summary
  MAC ACL mlist1
  Total rules configured: 6

  MAC ACL mlist2
  Total rules configured: 3

  MAC ACL mlist3
  Total rules configured: 1

  MAC ACL mlist4
  Total rules configured: 0
  ```
show route-map

The **show route-map** command displays the contents of configured route maps.

**Command Mode**
- EXEC

**Command Syntax**
- `show route-map [map_name]`

**Parameters**
- `<no parameter>` displays the content of all configured route maps
- `map_name` displays the content of the specified route map

**Example**
- This command displays the **map1** route map.
  ```
  switch(config)#show route-map map1
  route-map map1 permit 10
  Description:
  Match clauses:
  SubRouteMap:
  Set clauses:
    set as-path prepend last-as 12
    set as-path prepend auto auto
  ```

- This command displays the **map** route map.
  ```
  switch>show route-map map
  route-map map permit 5
  Match clauses:
    match as 456
  Set clauses:
  route-map map permit 10
  Match clauses:
    match ip next-hop 2.3.4.5
    match as-path path_2
  Set clauses:
    set local-preference 100
  ```
show platform fap acl

The **show platform fap acl** command displays the ACL information of Sand platform devices.

**Command Mode**

Privileged EXEC

**Command Syntax**

```
show platform fap acl [ipkgv | l4ops | mirroring | opkgv | pmf | tcam | udf | vsicfg]
```

**Parameters**

- **ipkgv** displays the ACL Ingress Interface Specification (IPKGV) information.
- **l4ops** displays the ACL Layer 4 Options (L4OPS) information.
- **mirroring** displays the mirroring ACL information.
- **opkgv** displays the ACL Egress Interface Specification (OPKGV) information.
- **pmf** displays the Pmf.
- **tcam** displays the ACL TCAM information.
- **udf** displays the ACL UDF information.
- **vsicfg** displays the ACL Virtual Switch Instance (VSI) CONFIG information.

**Guidelines**

This command is supported on DCS-7280SE and DCS-7500E series platforms only.

**Examples**

- This command displays the brief information of all installed mirroring ACLs.
  ```
  switch(config)#show platform fap acl mirroring
  ================
  Aggregate ACLs
  ================

  (list2:0->2) type=2; version=0
  - list2 [ prio 0 ] => session 2

  (list1:10->1,list3:20->3) type=0; version=13
  - list3 [ prio 20 ] => session 3
  - list1 [ prio 10 ] => session 1
  
  ===============
  Interface-ACL Mapping
  ===============

  Ethernet1 => (list1:10->1,list3:20->3) [ ipv4 ]
  Ethernet33 => (list2:0->2) [ mac ]
  ```
show platform fap acl tcam

The `show platform fap acl tcam` command displays the number of TCAM entries (hardware resources) occupied by the ACL on each forwarding ASIC of Sand platform devices.

**Command Mode**

Privileged EXEC

**Command Syntax**

```
show platform fap acl tcam [detail | diff | hw | shadow | summary]
```

**Parameter**

- **detail** displays the number of TCAM entries (hardware resources) occupied by the ACL on each forwarding ASIC.
- **diff** displays the difference between hardware and shadow.
- **hw** displays ACL entries from hardware.
- **shadow** displays ACL entries from shadow.
- **summary** displays the percentage of TCAM utilization per forwarding ASIC.

**Example**

- This command displays the number of TCAM entries and other ACL TCAM detail.

```
switch# show platform fap acl tcam detail
ip access-list ipAcl0000 (RACL, 1 rules, 2 entries, direction in, state success)
  Shared: false
  Interface: Vlan0002
   --------------------
  Fap: Arad3/0
  Bank Offset Entries
  1   0   2
  Interface: Vlan0003
   --------------------
  Fap: Arad3/0
  Bank Offset Entries
  1   2   2
  Fap: Arad3/4
  Bank Offset Entries
  1   0   2
```
show platform arad acl tcam

The `show platform arad acl tcam` command displays the number of TCAM entries (hardware resources) occupied by the ACL on each forwarding ASIC.

This command is applicable only on DCS-7500E, DCS-7280E series switches.

**Command Mode**

EXEC

**Command Syntax**

```
show platform arad acl tcam [scope]
```

**Parameters**

- `scope` specifies the information displayed. Options include:
  - `detail` Displays the ACL TCAM details
  - `diff` Displays the difference between hardware and shadow
  - `hw` Displays the ACL entries from hardware
  - `shadow` Displays the ACL entries from shadow
  - `summary` Displays the ACL TCAM summary

**Example**

- This command displays the number of TCAM entries used by Arad0 ASIC. In this example, ACL is applied on 2 VLANs (Vl2148 and Vl2700) but number of TCAM entries occupied is only 1.

```
switch# show platform arad acl tcam detail
ip access-list ipAclLimitTest (Shared RACL, 0 rules, 1 entries, direction out, state success, Acl Label 2)
Fap: Arad0, Shared: true, Interfaces: Vl2148, Vl2700
  Bank Offset Entries
  0   0   1
Fap: Arad1, Shared: true, Interfaces: Vl2148
  Bank Offset Entries
  0   0   1
```

- This command displays the percentage of TCAM utilization per forwarding ASIC.

```
switch# show platform arad acl tcam summary
The total number of TCAM lines per bank is 1024.

Arad0:

<table>
<thead>
<tr>
<th>Bank</th>
<th>Used</th>
<th>Used %</th>
<th>Used By</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>IP Egress PACLs/RACLs</td>
</tr>
</tbody>
</table>

Total Number of TCAM lines used is: 1

Arad1:

<table>
<thead>
<tr>
<th>Bank</th>
<th>Used</th>
<th>Used %</th>
<th>Used By</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>IP Egress PACLs/RACLs</td>
</tr>
</tbody>
</table>

Total Number of TCAM lines used is: 1
```
show platform arad acl tcam summary

The show platform arad acl tcam summary command displays the percentage of TCAM utilization per forwarding ASIC.

Command Mode

EXEC

Command Syntax

show platform arad acl tcam summary

Parameter

• summary displays the ACL TCAM summary.

Example

• This command displays the percentage of TCAM utilization per forwarding ASIC.

```
switch#show platform arad acl tcam summary
The total number of TCAM lines per bank is 1024.

========================================================================
Arad3/0:
========================================================================
Bank Used Used % Used By
1 4 0 IP RACLs
Total Number of TCAM lines used is: 4

========================================================================
Arad3/4:
========================================================================
Bank Used Used % Used By
1 2 0 IP RACLs
Total Number of TCAM lines used is: 2
```
show platform arad mapping

The `show platform arad mapping` command displays the mapping between the interfaces and the forwarding ASICS.

**Command Mode**

EXEC

**Command Syntax**

`show platform arad chip_name mapping`

**Parameter**

- `chip_name` specifies the Arad chip name.

**Example**

- This command displays the mapping between the interfaces and the forwarding ASICS on arad3/0 chip.

```snippet
switch#show platform arad arad3/0 mapping
Arad3/0 Port    SysPhyPort    Voq   (Fap,FapPort)    Xlge    Serdes
------------------------------------------------------------------
Ethernet3/1/1    34    288        (0 , 2)     n/a        (20)
```

```
**show platform trident tcam**

The `show platform trident tcam` command displays the TCAM entries configured for each TCAM group including policy maps and corresponding hits.

**Command Mode**

EXEC

**Command Syntax**

```
show platform trident tcam [acl | cpu-bound | detail | entry | mirror | pbr | pipe | qos | shared | summary]
```

**Parameters**

- `<no parameters>` displays TCAM entries for each TCAM group.
- `acl` displays the trident ACL information.
- `cpu-bound` displays the trident cpu-bound information.
- `detail` Lists all TCAM entries.
- `entry` displays the TCAM entry information.
- `mirror` displays the trident Mirroring ACL information.
- `pbr` displays the trident PBR ACL information.
- `pipe` allows to specify a pipe for filtering.
- `qos` displays the trident QOS information.
- `shared` displays the ACL Sharing information.
- `summary` displays the TCAM allocation information.

**Guidelines**

This command is applicable only on DCS-7010, DCS-7050/DCS-7050X, DCS7250X, DCS-7300X series switches.

**Examples**

- This command displays the trident mirroring ACL information.
  ```
switch(config)#show platform trident tcam mirror
  === Mirroring ACLs on switch Linecard0/0 ===
  Session: mir-sess2

  INGRESS ACL mirAc12* uses 2 entries
  Assigned to ports: Ethernet32/1
  ```
- This command displays detailed information for the TCAM group.

```plaintext
switch#show platform trident tcam detail
=== TCAM detail for switch Linecard0/0 ===
TCAM group 9 uses 42 entries and can use up to 1238 more.

Mlag control traffic uses 4 entries:
- 589826: 0 hits - MLAG - SrcPort UDP Entry
- 589827: 0 hits - MLAG - DstPort UDP Entry
- 589828: 0 hits - MLAG - SrcPort TCP Entry
- 589829: 0 hits - MLAG - DstPort TCP Entry

CVX traffic reserves 6 entries (0 used).
L3 Control Priority uses 23 entries:
- 589836: 0 hits - URM - SelfIp UDP Entry
- 589837: 0 hits - URM - SelfIp TCP Entry
- 589848: 0 hits - OSPF - unicast
- 589849: 71196 hits - OSPFv2 - Multicast
- 589850: 0 hits - OSPFv3 - Multicast
- 589851: 0 hits - OSPF Auth ESP - Multicast
- 589852: 0 hits - OSPF Auth ESP - Unicast
- 589853: 0 hits - IP packets with GRE type and ISIS protocol
- 589854: 0 hits - RouterL3 Vlan Priority 6,7 Elevator
- 589855: 0 hits - RouterL3 DSCP 48-63 Elevator
- 589856: 0 hits - RouterL3 Priority Elevator
- 589857: 0 hits - NextHopToCpu, Glean
- 589858: 0 hits - L3MC Cpu OIF

IGMP Snooping Flooding reserves 8 entries (6 used):
- 589864: 0 hits - IGMP Snooping Restricted Flooding L3 from local mlag peer
- 589865: 0 hits - IGMP Snooping Restricted Flooding L3

L4 MicroBfd traffic reserves 1 entries (0 used).
TCAM group 13 uses 99 entries and can use up to 1181 more.

Dot1x MAB traffic uses 1 entries:
- 851968: 0 hits - Dot1xMab Rule

<-------OUTPUT OMMITED FROM EXAMPLE-------->

ck338.22:14:38(config-pmap-qos-policy1)#
```
show hardware tcam profile

The `show hardware tcam profile` command displays the hardware specific information for the current operational TCAM profile in the running configuration.

This command is applicable only on DCS-7280(E/R), DCS-7500(E/R) series switches.

**Command Mode**

`EXEC`

**Command Syntax**

```
show hardware tcam profile
```

**Parameters**

- `tcam` specifies the TCAM information.
- `profile` specifies the TCAM profile information.

**Example**

- This command displays the current operational TCAM profile details.

```
switch#show hardware tcam profile
Configuration        Status
FixedSystem          default              default
```
show platform fap acl tcam hw

The show platform fap acl tcam hw command displays the TCAM entries configured for each TCAM bank including policy-maps and corresponding traffic match.

This command is applicable only on DCS-7280(E/R), DCS-7500(E/R) series switches.

**Command Mode**

EXEC

**Command Syntax**

```
show platform fap fap_name acl tcam hw
```

**Parameters**

- `fap_name` specifies the switch chip-set name.
- `acl` specifies the Arad ACL information.
- `tcam` specifies the Arad TCAM information.
- `hw` specifies the ACL entries for hardware.

**Example**

- This command displays the TCAM entries configured for each TCAM bank including policy maps and corresponding traffic matches.

```
switch#show platform fap Arad1 acl tcam hw
```

---

```
Arad1 Bank 0 Type: dbPdpIp, dbPdpIp6, dbPdpMpls, dbPdpNonIp, dbPdpTunnel

<table>
<thead>
<tr>
<th>Offs</th>
<th>X</th>
<th>PR</th>
<th>TT</th>
<th>R</th>
<th>QI</th>
<th>V6MC</th>
<th>DPRT</th>
<th>SPRT</th>
<th>F</th>
<th>DEST</th>
<th>V</th>
<th>ACT</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>4</td>
<td>59</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>4</td>
<td>33</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>00000</td>
<td>0</td>
</tr>
<tr>
<td>31</td>
<td>4</td>
<td>32</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0008f</td>
<td>0</td>
</tr>
<tr>
<td>32</td>
<td>4</td>
<td>06</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>00000</td>
<td>0</td>
</tr>
<tr>
<td>33</td>
<td>4</td>
<td>10</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>00097</td>
<td>0</td>
</tr>
<tr>
<td>34</td>
<td>4</td>
<td>06</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>00000</td>
<td>0</td>
</tr>
</tbody>
</table>

Arad1 Bank 1 Type: dbIpQos

<table>
<thead>
<tr>
<th>Offs</th>
<th>X</th>
<th>T0</th>
<th>QI</th>
<th>FOI</th>
<th>TT1</th>
<th>DEST</th>
<th>TT1P</th>
<th>PT</th>
<th>VX_DP</th>
<th>PN</th>
<th>F</th>
<th>MC</th>
<th>V</th>
<th>HDR OFFSETS</th>
<th>ACT</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td>3</td>
<td>00000</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td></td>
<td>01</td>
<td></td>
<td>0</td>
<td>00000</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<--------OUTPUT OMITTED FROM EXAMPLE-------->

---
system profile

The system profile command creates a new Ternary Content-Addressable Memory (TCAM) profile in the running configuration.

The default system profile and no system profile commands delete non-default TCAM profiles from the running configuration.

Command Mode

Hardware TCAM

Command Syntax

```
system profile [profile_name | default | mirroring-acl | pbr-match-nexthop-group
| qos | tap-aggregation-default | tap-aggregation-extended | tc-counters]
default system profile
no system profile
```

Parameters

- `profile_name` creates a profile with the specified name.
- `default` creates a default profile.
- `mirroring-acl` creates a mirroring-ACL profile.
- `pbr-match-nexthop-group` creates a pbr-match-nexthop-group profile.
- `qos` creates a Quality of Service (QoS) profile.
- `tap-aggregation-default` creates a tap-aggregation-default profile.
- `tap-aggregation-extended` creates a tap-aggregation-extended profile.
- `tc-counters` creates a tc-counters profile.

Guideline

These commands are compatible with the DCS-7280SE and DCS-7500E series switches only.

Examples

- These commands create a mirroring-ACL profile.

```
switch(config)#hardware tcam
switch(config-hw-tcam)#system profile mirroring-acl
switch(config-hw-tcam)#show hardware tcam profile
  Configuration  Status
  FixedSystem    mirroring-acl    mirroring-acl
switch(config-hw-tcam)#
```
• These commands delete non-default TCAM profiles.

```plaintext
switch(config)#hardware tcam
switch(config-hw-tcam)#show hardware tcam profile

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linecard9</td>
<td>mirroring-acl</td>
</tr>
<tr>
<td>Linecard8</td>
<td>mirroring-acl</td>
</tr>
<tr>
<td>Linecard3</td>
<td>mirroring-acl</td>
</tr>
<tr>
<td>Linecard4</td>
<td>mirroring-acl</td>
</tr>
<tr>
<td>Linecard6</td>
<td>mirroring-acl</td>
</tr>
</tbody>
</table>

switch(config-hw-tcam)#default system profile
switch(config-hw-tcam)#show hardware tcam profile

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linecard9</td>
<td>default</td>
</tr>
<tr>
<td>Linecard8</td>
<td>default</td>
</tr>
<tr>
<td>Linecard3</td>
<td>default</td>
</tr>
<tr>
<td>Linecard4</td>
<td>default</td>
</tr>
<tr>
<td>Linecard6</td>
<td>default</td>
</tr>
</tbody>
</table>

switch(config-hw-tcam)#
```

• These commands delete TCAM profiles.

```plaintext
switch(config-hw-tcam)#show hardware tcam profile

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linecard9</td>
<td>tc-counters</td>
</tr>
<tr>
<td>Linecard8</td>
<td>tc-counters</td>
</tr>
<tr>
<td>Linecard3</td>
<td>tc-counters</td>
</tr>
<tr>
<td>Linecard4</td>
<td>tc-counters</td>
</tr>
<tr>
<td>Linecard6</td>
<td>tc-counters</td>
</tr>
</tbody>
</table>

switch(config-hw-tcam)#no system profile
switch(config-hw-tcam)#show hardware tcam profile

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linecard9</td>
<td>default</td>
</tr>
<tr>
<td>Linecard8</td>
<td>default</td>
</tr>
<tr>
<td>Linecard3</td>
<td>default</td>
</tr>
<tr>
<td>Linecard4</td>
<td>default</td>
</tr>
<tr>
<td>Linecard6</td>
<td>default</td>
</tr>
</tbody>
</table>

switch(config-hw-tcam)#
```
hardware access-list resource sharing vlan in

The **hardware access-list resource sharing vlan in** command enables the IPv4 Ingress Sharing of hardware resources on the switch same ACL is applied on different VLANs.

The **no hardware access-list resource sharing vlan in** command disables the IPv4 Ingress Sharing of hardware resources on the switch.

**Command Mode**
- Global Configuration

**Command Syntax**
- `hardware access-list resource sharing vlan in`
- `no hardware access-list resource sharing vlan in`

**Guideline**
- This command is compatible only with the DCS-7010 and DCS-7050x series switches.
- Enabling IPv4 Ingress Sharing requires the restart of software agents on the platform, this is a disruptive process and will impact traffic forwarding.
  - Use `show platform trident` command to verify the Ingress IPv4 Sharing information.
hardware access-list resource sharing vlan ipv4 out

The **hardware access-list resource sharing vlan ipv4 out** command enables the IPv4 Egress RACL TCAM sharing on the switch.

The **no hardware access-list resource sharing vlan ipv4 out** command disables the IPv4 Egress RACL TCAM sharing on the switch. By default, the IPv4 Egress RACL sharing is enabled on the switch.

**Command Mode**
- Global Configuration

**Command Syntax**

```
hardware access-list resource sharing vlan ipv4 out
no hardware access-list resource sharing vlan ipv4 out
```

**Guideline**
- This command is compatible only with the DCS-7280E and DCS-7500E series switches.
- Disabling IPv4 RACL sharing requires the restart of software agents on the platform. This is a disruptive process and will impact traffic forwarding.
- Enabling IPv4 RACL sharing, if previously disabled from the default configuration, requires the restart of software agents on the platform. This is a disruptive process and will impact traffic forwarding. Enabling IPv4 RACL sharing if uRPF is configured will disable uRPF.
- Use **show running-config all | include sharing** command to verify whether or not sharing for egress IPv4 RACLs is enabled.

**Example**
- This command verifies whether IPv4 RACL sharing is enabled or disabled.

```
switch# show running-config all | include sharing
  hardware access-list resource sharing vlan ipv4 out ----> It returns the following output if IPv4 RACL sharing is enabled.
```